

**DIINS FAILURE DETECTION ISOLATION AND
RECONFIGURATION PROCEDURES (U)**

by
D.A.G. Arden

DALE ARDEN CONSULTING

REPORT NO. DAC-03-97

**August 1997
Ottawa**

EXECUTIVE SUMMARY

The Dual Inertial Integrated Navigation System (DIINS) is a Kalman filter based, multi-sensor, optimally integrated navigation system developed at the Defence Research Establishment Ottawa. Development of DIINS was initiated with a view to satisfying the navigational requirements of the Canadian Navy's TRUMP, CPF and future submarines. DIINS has been designed to provide a single source of navigation information (position, velocity, attitude etc.) to other ship board systems with minimize data latency, improve navigation reliability and accuracy, and provide automatic sensor failure detection.

DIINS differs from earlier integrated navigation systems developed at DREO in two main areas: it is based on (two) inertial navigation systems; and it was designed to have a number of filters running concurrently.

The specific architecture selected for DIINS has been named a Hierarchic Alliance of Filters. It consists of several independently operating Kalman filters arranged in a "root" structure. Each of two main branches is based on data from one INS and consists of a number of filter levels. At the top is the "unifilter", which uses all available aiding sensors. Filters in subsequent levels exclude various combinations of aiding sensors in an arrangement that ensures there is always at least one uncorrupted filter, regardless of the sensor failure status.

The highest level filter which is safe from corruption by a sensor failure is eligible for use as DIINS' output source. In most circumstances, one of the unifilters is used. If an aiding sensor failure occurs which has the potential to corrupt a unifilter, the best filter in the next lower branch is selected as the output source.

This report begins by describing two types of sensor failures — hard (an abrupt change in sensor performance identified by measurement tests) and soft (a slowly accumulating change in sensor performance identified by specialized tests). Methods of detecting both failure types are described. Then, procedures are derived which combine the failure detection tests from more than one filter to isolate sensor failures. Finally, failure detection and isolation results are used to select the best source for DIINS' output. This is normally a filter but may be a particular sensor in extreme situations. Cross-references to DIINS software files and some post-processed FDIR results are included in the appendices.

Contents

EXECUTIVE SUMMARY	ii
List of Tables	v
List of Figures	vi
Chapter 1 INTRODUCTION	1
1.1 A Brief Description of DIINS	1
1.2 DIINS FDIR	2
Chapter 2 HARD FAILURES	4
2.1 Hard Failure Detection	5
2.1.1 Count of Formed Measurements	5
2.1.2 Count of Failed Measurements	6
2.1.3 The Kalman Filter Residual Test	6
2.2 Hard Failure Isolation	9
2.2.1 Selection of Hard Failure Isolation Filters	11
Chapter 3 SOFT FAILURES	17
3.1 Soft Failure Detection	17
3.1.1 The Theoretical χ^2 Soft Failure Detection Test	17
3.1.2 The DIINS χ^2 Soft Failure Detection Test	19
3.2 Soft Failure Isolation	21
3.2.1 Selection of Soft Failure Isolation Filters	22
Chapter 4 SELECTION OF OPTIMAL SOURCE FOR OUTPUT	27
4.1 Filter Classification	29
4.1.1 Selection of the Best Filter for Output	30
4.1.2 Selection of the Best Sensor for Output	31
4.1.3 How Many Filters?	32
4.2 Measurement Reconfiguration	33
4.2.1 GPS Measurement Groups	35
4.2.2 Loran-C Measurement Groups	37
4.2.3 INS Measurement Groups	40

Chapter 5	CONCLUSIONS	41
References		42
Appendix A	DIINS FDIR SOFTWARE CROSS-REFERENCES	43
A.1	Hard Failure Detection and Level 1 and 2 Measurement Reconfiguration	43
A.2	Soft Failure Detection	45
A.3	Hard Failure Isolation	46
A.4	Soft Failure Isolation	46
A.5	FDIR Reconfiguration	47
A.6	Level 3 Measurement Reconfiguration	47
Appendix B	TEST RESULTS	49
B.1	Output Log for the INS1+INS2+GPS+Loran-C, 7 Filter Configuration	50
B.2	Output Log for the INS1+INS2+GPS, 3 Filter Configuration . . .	59
B.3	Output Log for the INS1+INS2+Loran-C, 3 Filter Configuration .	63
B.4	Output Log for the INS1+INS2, 1 Filter Configuration	67
B.5	Output Log for the INS1+GPS, 1 Filter Configuration	69
B.6	Output Log for the INS1+Loran-C, 1 Filter Configuration	71
B.7	Output Log for the INS2+GPS, 1 Filter Configuration	73
B.8	Output Log for the INS2+Loran-C, 1 Filter Configuration	75
Appendix C	ADDING DIFFERENTIAL GPS AS A THIRD AIDING SENSOR	77

List of Tables

Table 2.1	Two-Tailed Standard Normal Test Equivalency	8
Table 2.2	"No Fail" Hard Sensor Failure Isolation	14
Table 2.3	Hard Sensor Failure Isolation After an INS1 Soft Failure	14
Table 2.4	Hard Sensor Failure Isolation After an INS2 Soft Failure	15
Table 2.5	Hard Sensor Failure Isolation After an Aid1 Soft Failure	15
Table 2.6	Hard Sensor Failure Isolation After an Aid2 Soft Failure	15
Table 2.7	Hard Sensor Failure Isolation After a χ^2 Failure of the INS1 + INS2 Filter	16
Table 3.1	"No Fail" Soft Sensor Failure Isolation	24
Table 3.2	Soft Sensor Failure Isolation After an INS1 Hard Failure	25
Table 3.3	Soft Sensor Failure Isolation After an INS2 Hard Failure	25
Table 3.4	Soft Sensor Failure Isolation After an Aid1 Hard Failure	26
Table 3.5	Soft Sensor Failure Isolation After an Aid2 Hard Failure	26
Table 4.1	Minimum GPS Dependencies	36
Table 4.2	Maximum GPS Dependencies	36
Table 4.3	Measurement Reconfiguration for the TASMAN GPS Receiver	37
Table 4.4	Measurement Reconfiguration for an INS	40

List of Figures

Figure 2.1	Standard DIINS Filter Configuration in Terms of Sensors	13
Figure 2.2	Standard DIINS Filter Configuration in Terms of Filter Numbers .	14

1.1 A Brief Description of DIINS

The Dual Inertial Integrated Navigation System (DIINS) is a Kalman filter based, multi-sensor, optimally integrated navigation system developed at the Defence Research Establishment Ottawa. Development of DIINS was initiated with a view to satisfying the navigational requirements of the Canadian Navy's TRUMP, CPF and future submarines. The primary objectives of DIINS are to:

1. provide single source of navigation information (position, velocity, attitude etc.),
2. improve navigation reliability,
3. improve navigation accuracy,
4. minimize data latency to other ship board systems,
5. provide automatic sensor failure detection.

all within the constraints of the available sensor suite.

DIINS follows on earlier work at DREO:

1. MINS (Marine Integrated Navigation System) is the first generation marine optimally integrated navigation system developed at DREO. It was based on a gyrocompass, speedlog and the radionavigation sensors available when MINS was developed. MINS was installed on many Navy ship.
2. PLANS (Primary Land Arctic Navigation System) is an optimally integrated navigation system designed and developed at DREO to work onboard a land vehicle at high latitudes.

DIINS differs from these earlier systems in two main areas:

1. it is based on (two) inertial navigation systems,
2. and it was designed to have a number of filters running concurrently.

The first change required the analysis and development of fairly complex inertial error models. The second change was imposed by the requirement to maximize the reliability of the navigation system. It was also aided by the extra redundancy supplied by having two INSs available.

The specific architecture selected for DIINS is being called a Hierarchic Alliance of Filters. It consists of several independently operating Kalman filters arranged in a "root" structure. At the top of the structure is a filter which combines data from each of the INSs. This filter is used exclusively for failure isolation of the INSs. Directly below this filter are two "unifilters", each based on one INS and using measurements from all available aiding sensors. Branching out from each unfilter are n additional filters (where n is the number of aiding sensors), each of which excludes measurements from one aid. The next lower level of branching is composed of the set of filters which each exclude two aiding sensors. The lowest level of branching contains n filters which each use one of aids. With this architecture, there are n levels (in two branches) below the INS1/INS2 filter.

In most circumstances, output from one of the unifilters is used for navigation output. If an aiding sensor failure occurs which has the potential to corrupt a unfilter, the filter in the

next lower branch which does not use the failed sensor is selected as the new output source. Failure isolation of an INS removes that INS' branch of filters from consideration for output: the best filter in the healthy INS' branch would be selected. The corruptible failure of a second aiding sensor would lead to the filter in the next lower level which excludes both failed sensors. If the second INS fails, unfiltered data from the best aiding sensor would be used for output. Recovery of a failed sensor can result in the reversal of this output selection procedure. This architecture provides an optimal uncorrupted filter under most failure conditions, and unfiltered aiding sensor output if both INSs have failed.

Details of the DIINS Kalman filter algorithms and error models can be found in reference [1]. Real time results from the 1995 sea trials are described in reference [2].

1.2 DIINS FDIR

Failure Detection, Isolation and Reconfiguration (FDIR) is the name given to techniques which try to preserve the integrity of a system of Kalman filters by minimizing the effects of failed or failing components. In an integrated navigation system such as DIINS, the components being scrutinized are the navigation sensors. There are several important reasons for employing FDI procedures:

1. The operator can be notified that a sensor is behaving in an unexpected manner. This makes the operator aware of a possible degradation in performance. In addition, if the operator knows that a sensor has failed, he may be able to alleviate the problem by repairing faulty cables, replacing failed components, etc.
2. Knowledge of a sensor failure can be passed on to filters which are using the sensor but may not have any internal indication of problems. Then, measurements which would have been made with the failed sensor, and may have corrupted the filter, can be disabled.
3. Knowledge of current sensor status helps in the identification and selection of the best source for output.

In general, a sensor "failure" can be defined as an error which was not anticipated in the Kalman filter modelling. In a system as critical as navigation, it is important that all possible sensor failures be identified. Sensor failures are separated into hard and soft types with distinct detection and isolation algorithms in DIINS. Failure detection takes place in individual filters; failure isolation is performed by the FDI processor using more than one filter. DIINS currently allows up to two INS's (the primary sensors), two GPS receivers supplying any combination of position, velocity and attitude, and one Loran-C receiver supplying time delays. The addition of aiding sensors adds to the FDIR complexity as was shown with the recent addition of the second (differential) GPS receiver. Appendix C outlines some of issues considered before dGPS was added.

Hard failures are sudden and significant. They may be caused by data problems (eg. total sensor failure, bad status from Built In Tests, data gaps, etc.); they may result from large, unexpected increases in process or measurement noise (eg. EM jamming or loss of signal strength); or, they may result from large, unexpected systematic errors (eg. a Loran-C cycle slip).

Soft failures are characterized by slowly degrading sensor performance (eg. a slow increase in the bias of an accelerometer). If the errors do not grow fast enough to trigger a hard failure, the filter may force the failure into inappropriate error states. In fact, a soft failure is perhaps most accurately defined as the result of unmodelled errors which are large enough to corrupt filtered results but cannot be identified by hard failure detection techniques.

Hard failure detection is separated into two tests, one to empirically test for the presence of usable sensor data and another to statistically test the quality of each measurement after it has been formed. Hard failure detection in a particular filter will result in the disabling of all dependent measurements in that filter to minimize any possible corruption arising from the failure. Hard failure isolation will result in the disabling of all measurements using the failed sensor in all filters. A sensor which is suffering from a soft failure will not be tested for hard failure isolation.

Soft failure detection acts on the complete filter by applying a statistical test to a subset of the Kalman filter state vector. If a soft failure is detected, all sensors used in the filter are flagged. The soft failure detection results from more than one filter are analyzed for soft (sensor) failure isolation. However, unlike hard failures, soft failure isolation does **not** have any effect on filter measurements. Rather, the soft failure results (along with hard failure results and other status indicators) are used in the selection of the most reliable output. A sensor which is suffering from a hard failure will not be tested for soft failure isolation.

All the FDIR procedures used in DIINS are described below. A theoretical description of the DIINS FDIR concepts is provided in reference [3].

In DIINS, there are two types of hard failures. Both are related to the status of the Kalman filter measurements. In brief, a sensor hard failure will be declared when

1. there is and has been no data from the sensor which could be used to form Kalman filter measurements;
2. or, measurements have been formed using the sensor's data, but the resulting residuals have been judged to be outside statistically acceptable bounds.

In both cases, failure detection relies on software counters residing in individual filters. At the start of each FDI cycle, all FDI counters are zeroed. Then, they are incremented in response to failure detection conditions. At the end of the FDI cycle, the counters are sent to the FDI processor where failure isolation algorithms combine the counters from selected filters to try to identify sensor failures.

The most basic unit employed in hard failure processing is the Kalman filter measurement. Each measurement is formed using data from two sensors, the primary INS and an aiding sensor (eg. INS1 and GPS). A failure condition in either sensor will affect the measurement. A statistical test of the difference between the measurement and a filter predicted estimate of the measurement is used to identify individual measurement failures. However, a failure of this so-called "residual test", cannot identify which contributing sensor is causing the failure. For this reason, the hard failure counters for the primary INS and the aiding sensor are always incremented together.

There are two hard failure detection and isolation counters for every activated sensor: one counter is incremented every time a measurement is formed; the other is incremented every time a measurement failure is detected. The counters, conditions under which they are incremented, and the pertinent FDI algorithms are described below.

Measurements are collected into various sets in the hard failure algorithms discussed below. The sets are defined here for reference:

1. Measurement Type — the set of all measurements formed using a particular parameter from a particular aiding sensor, eg. GPS latitude measurements.
2. Measurement Group — the set of similar measurement Types from a particular aiding sensor, eg. GPS position measurements. (This is used in Section 4.2.)
3. Measurement Class — the set of all measurements from a particular aiding sensor, eg. GPS position, velocity and attitude measurements.

Currently, DIINS can process the following Groups of measurements:

1. GPS
 - a. positions,
 - b. velocities,
 - c. attitude,
 - d. pseudoranges. *

2. INS

- a. positions,
- b. velocities,
- c. attitude.

3. Loran-C

- a. time differences.

4. Omega

- a. range differences. *

* — not fully implemented or tested.

In DIINS, each Group of measurements is processed in a separate software procedure. However, members of a measurement Class may be formed in different procedures. For example, if GPS is an aiding sensor in a filter, there are three separate procedures for forming position, velocity, and attitude measurements. However, all GPS position, velocity, and attitude measurements belong to the same (GPS) measurement Class.

2.1 Hard Failure Detection

Count of Formed Measurements

Before sensor data is made available to form measurements in the Kalman filter, it is tested for usability. A sensor will be judged **not** usable whenever ANY one of three conditions is identified:

1. When the sensor status flag is set to BAD. Status flags are set by sensor Built In Tests (BITs) or by DIINS' data collection tasks (DCT). This status flag will be called the BIT/DCT status flag in the following discussions. For example, when there are time gaps in sensor data, DIINS will fill in the gaps with any convenient data records and then identify the record as unusable by setting its status flag to the failure condition. If the status of the primary INS is bad, the FDI processor is immediately notified: no filter using an INS with bad status should be used for output.
2. When the most recent data from the sensor has previously been used to form measurements. (Data from different sensors will typically be received at different rates.)
3. When the difference in tagged times for the primary INS and the aiding sensor is judged to be too large to allow safe extrapolation. Measurements are formed by comparing similar parameters from two sensors. In general, the two parameters will refer to different points in time requiring that one sensor's data be extrapolated to the time tag of the second sensor.

A measurement Class was defined above as that set of all measurements formed using a specific aiding sensor (eg. GPS). If the primary INS and aiding sensor used to form a particular measurement Class are both judged to be usable according to the three tests listed above, a "*MeasFormed*" counter for each sensor is incremented.

Thus, for any one filter and one measurement Class, assuming the INS and aiding sensor are both usable, the MeasFormed counter for the aiding sensor is incremented by one for every measurement cycle in one FDI cycle. On the other hand, since the INS is used in every measurement, the MeasFormed counter for the INS would be incremented by the number of usable aiding sensors for every measurement cycle in one FDI cycle. For example, if the filter uses one aid (eg. GPS), and the FDI cycle time is 10 seconds while the measurement cycle time is 5 seconds, MeasFormed(INS) and MeasFormed(Aid) would be incremented by one (one aid) twice (2 epochs) — MeasFormed(INS) = 2 and MeasFormed(GPS) = 2 would be sent to the FDI processor. If the filter uses two aids (eg. GPS and Loran-C), and all three sensors are usable, with the same 10 and 5 second cycle times, MeasFormed(INS) would be incremented by two (2 aids) twice (2 epochs) — MeasFormed(INS) = 4 and MeasFormed(GPS) = MeasFormed(Loran-C) = 2 would be sent to the FDI processor.

For a filter in steady state, the MeasFormed counters are constant. The counters change only in response to changes in sensor performance. To filter out transient changes (such as one or a few bad data records), changes in the MeasFormed counters are sent to the FDIR processor only if the change is persistent. A persistent change is specified by a preset time interval.

Count of Failed Measurements

The next level of hard failure detection starts only after

1. the INS and aiding sensor have been judged usable,
2. the MeasFormed counters have been incremented, and
3. the data has been forwarded to the appropriate measurement procedure.

In brief, measurement failures are identified in three steps:

1. Every time a measurement is formed, its Kalman filter residual is computed and then tested against statistically expected limits. If a particular residual fails this test, it is prevented from updating the Kalman state estimates.
2. A measurement failure is declared after a particular measurement has failed its residual test X percent of the time over the last Y seconds (eg. 90% failures over the last 5 minutes).
3. After a measurement has been declared 'FAILED',
 - a. The failed measurement will not be used to update the filter until its recovery is assured (as described in Section 2.2 below).
 - b. The "*MeasFailed*" counters for the INS and the aiding sensor involved (eg. INS1 and GPS) are incremented. The MeasFailed counters are sent to the FDI processor for hard sensor failure isolation.

The basis of this failure detection scheme, the residual test, deserves at bit more attention: it will be described in more detail below.

The Kalman Filter Residual Test

A linear Kalman filter residual is defined as

$$\vec{v} = \vec{z} - H \hat{\vec{x}}(-) \quad (2.1)$$

(see eg. reference [4], pp. 107-108) where

- $\tilde{z}$, the Kalman measurement vector, is defined as

$$\tilde{z} = H\tilde{x} + \tilde{\epsilon} \quad (2.2)$$

For a specific parameter ρ , the measurement is computed as

$$z_\rho = \hat{\rho}(\text{Aid}) - \hat{\rho}(\text{INS}) \quad (2.3)$$

- $\hat{x}(-)$ is the predicted Kalman state vector before measurement update;
- H is the known measurement matrix expressing the linear relationship between the measurements and the state vector in the absence of measurement error;
- $\tilde{\epsilon}$ is the measurement error vector.

All error terms are defined as "true" minus approximate so that the error state for some parameter ρ is

$$x_\rho = \bar{\rho} = \rho - \hat{\rho} \quad (2.4)$$

Similarly, the measurement equation 2.3 can be considered to be the "true" Aid value minus the "approximate" INS value. (In this case, the "true" Aid value is just more accurate than the "approximate" INS value.)

The residual covariance matrix is

$$\begin{aligned} N &= E\{\tilde{v}\tilde{v}^T\} \\ &= E\left\{\left(\tilde{z} - H\hat{x}(-)\right)\left(\tilde{z} - H\hat{x}(-)\right)^T\right\} \\ &= E\left\{\left(H\tilde{x} + \tilde{\epsilon} - H\hat{x}(-)\right)\left(H\tilde{x} + \tilde{\epsilon} - H\hat{x}(-)\right)^T\right\} \\ &= E\left\{\left(H\tilde{x}(-) + \tilde{\epsilon}\right)\left(H\tilde{x}(-) + \tilde{\epsilon}\right)^T\right\} \\ &= H E\left\{\tilde{x}(-)\tilde{x}^T(-)\right\} H + E\left\{H\tilde{x}(-)\tilde{\epsilon}\right\} + E\left\{\tilde{\epsilon}\tilde{x}^T(-)H^T\right\} + E\left\{\tilde{\epsilon}\tilde{\epsilon}^T\right\} \end{aligned} \quad (2.5)$$

Assuming the measurement errors are all uncorrelated, $E\{\tilde{x}\tilde{\epsilon}^T\} = E\{\tilde{\epsilon}\tilde{x}^T\} = 0$ and

$$\begin{aligned} N &= H E\left\{\tilde{x}(-)\tilde{x}^T(-)\right\} H + E\left\{\tilde{\epsilon}\tilde{\epsilon}^T\right\} \\ &= H P(-) H^T + R \end{aligned} \quad (2.6)$$

Residuals are tested individually. In practice, an empirical two-tailed hypothesis of the form

$$|\nu_\rho| \leq m\sigma_\nu \quad (2.7)$$

is tested. In this scalar equation,

$$\begin{aligned} \nu_\rho &= z_\rho - H_\rho \hat{x}(-) \\ \sigma_\nu^2 &= H_\rho P(-) H_\rho^T + \sigma_n^2 \end{aligned} \quad (2.8)$$

and m is an empirically selected integer with typical values around 3 or 5. A residual failure occurs whenever the above hypothesis fails, ie. whenever

$$|\nu_p| > m\sigma_\nu \quad (2.9)$$

This empirical test can be explained in statistical terms. First of all, assume that the errors in the state estimates and measurements are unbiased and normally distributed:

$$\begin{aligned} \tilde{x} &\rightarrow n(\vec{0}, P) \\ \tilde{e} &\rightarrow n(\vec{0}, R) \end{aligned} \quad (2.10)$$

Under these conditions,

$$\tilde{\nu} \rightarrow n(\vec{0}, N) \quad (2.11)$$

The hypothesis that a particular residual belongs to this distribution, at a significance level α (or confidence level of $(1 - \alpha)$), is tested using

$$|\nu_p| \leq n_{(1-\alpha/2)}\sigma_\nu \quad (2.12)$$

(The Kalman filter process requires that the measurements be uncorrelated — N must be diagonal. The residual standard deviation, σ_ν , is the square root of the appropriate diagonal element of N . The empirical test of equation 2.9 can be considered a normal test at a confidence level of $(1 - \alpha)$ if $m = n_{(1-\alpha/2)}$. The following table provides a few examples of the equivalences:

Table 2.1 Two-Tailed Standard Normal Test Equivalency

$m = n_{(1-\alpha/2)}$	$(1-\alpha)$ Confidence Level
1	0.6827
2	0.9545
3	0.9973
5	>0.99995

In practice, the 2-sigma test is often referred to as a 95% test, and the 3-sigma test is often referred to as a 99% test.

Using the normal test table, the m -sigma residual test can be interpreted as follows: the fraction of residuals which are expected to pass an m -sigma test is equal to $(1 - \alpha)$. Conversely, a fraction equal to α , is expected to fail.

A 5-sigma test is often used in Kalman filtering practice even though, according to the table, there is a very low theoretical probability of a residual failing the test. Higher m -tests are typically required to prevent excessive numbers of failure false alarms. The false alarms occur because the residuals are not exactly normal. And, the residuals are not exactly normal because of practical limits imposed on the error modelling: small systematic sensor errors are not explicitly modelled. These systematic errors end up in $\tilde{x}$ or $\tilde{e}$ or both. The assumption of normality is partially violated leading to a non-normal distribution for the residuals. However, the residuals should always be "near normal": if they are not, the modelling should be improved. Also, some analytical insight is gained by thinking in terms of normally distributed residuals.

2.2 Hard Failure Isolation

Hard failure isolation (HFI) requires that the hard failure detection results from more than one filter be analyzed. For each sensor, a group of *hard failure isolation filters* is selected. (The selection of these filters requires some analysis which is presented below.) A hard sensor failure isolation is declared only when all HFI filters signal the failure. Recovery from a hard sensor failure is declared as soon as any HFI filter recovers.

Hard failure isolation occurs in two steps: first there is a test to see if there have been any measurements formed using the sensor being tested (this will be called the *TYPE M* hard sensor failure test). Then, if there have been measurements formed, another test determines whether any of those measurements were acceptable (this will be called the *TYPE A* hard sensor failure test).

Specifically, to isolate hard sensor failures, at each FDI epoch all measurement counters from all filters are sent to the FDI processor. Then, for each sensor (K),

1. The MeasFormed and MeasFailed counters from the preselected HFI filters are summed to calculate the "*TotalMeasFormed*" and "*TotalMeasFailed*" counters, respectively.
2. If the TotalMeasFormed counter is zero, there have been no measurements of the sensor K Class formed in any HFI filter: a TYPE M hard failure is signalled. In the software, the 'MEAS_PROCESSED' sensor status flag for the sensor is set to FALSE, and a message of the form "FDI: NO measurements are being processed from sensor K . . ." is sent to the operator. Topic 1 describes the possible causes of this failure condition.
3. If the TotalMeasFormed counter is greater than zero, proceed to the TYPE A test:
 - If the TotalMeasFormed counter is equal to the TotalMeasFailed counter, a measurement failure has been detected in every sensor K HFI filter: a TYPE A hard failure is declared. In the software, the 'HARD_FAILURE' sensor status flag is set to TRUE, and a message of the form "FDI: Sensor K HARD FAILURE . . ." is sent to the operator.
 - If the TotalMeasFailed counter is greater than zero but less than TotalMeasFormed, it is assumed that the sensor has not suffered (or has recovered from) a hard failure.

Recovery from both types of hard failure is allowed:

1. A TYPE M hard failure will be cancelled as soon as the TotalMeasFormed counter exceeds zero. This indicates that in at least one HFI filter, measurements in the sensor K Class were formed. It says nothing regarding the quality of those measurements. The 'MEAS_PROCESSED' flag is set to TRUE, and a message of the form "FDI: STARTED processing measurements from sensor K . . ." is sent to the operator.
2. A TYPE A hard failure will be cancelled as soon as the TotalMeasFormed counter exceeds the TotalMeasFailed counter. This indicates that in at least one HFI filter no sensor K measurement Class failure was detected — either an earlier failure was cancelled or a filter with a failure was replaced by a filter with no failure. The former is expected because the filters which should be most sensitive are selected for HFI. The 'HARD_FAILURE' flag for the sensor is set to FALSE and a message of the form "FDI: Sensor K RECOVERY from hard failure . . ." is sent to the operator.

Examples may make these processes easier to follow.

1. Assume that a GPS receiver was accidentally turned off.
 - a. DIINS will almost immediately recognize a problem and begin sending dummy GPS data with bad status flags to the filters.
 - b. The filters will start a timer when the status goes bad.
 - c. As soon as the bad status timer reaches its predefined limit, each filter will notify the central FDIR processor.
 - d. Since all filters receive the same bad status data, they will all signal the failure at the same time and the FDIR processor will isolate the TYPE M GPS failure.
 - e. The FDIR processor notifies the operator, who identifies the problem and turns the receiver back on.
 - f. As soon as the receiver is transmitting good data, DIINS will send it on to the filters.
 - g. And the failure isolation process is reversed:
 - a good status timer starts;
 - after a predefined delay, the FDIR processor is notified of the recovery;
 - the FDIR processor identifies the GPS recovery and notifies the operator.
2. Assume that a GPS receiver for some reason starts sending east longitude instead of west longitude but does not internally recognize the problem.
 - a. The bad GPS longitude is sent to the filters with good status flags.
 - b. The residuals for the longitude measurements are computed and will immediately fail.
 - c. As soon as X% of the longitude residuals fail in a period of Y seconds, the Meas-Failed(GPS) counter is incremented.
 - d. The FDIR processor will isolate the TYPE A GPS failure and notify the operator who can attempt to fix the error (eg. by reinitializing the receiver).
 - e. As soon as the correct longitude is available, the residuals should start to pass.
 - f. And the failure isolation process is reversed:
 - after a XX% of the residuals pass in YY seconds, the FDIR processor is notified of the recovery;
 - the FDIR processor identifies the GPS recovery and notifies the operator.

A couple of other points to consider in relation to DIINS' hard failure isolation processing:

1. Hard failure isolation will NOT be declared for a sensor
 - a. which is currently suffering from a soft failure;
 - b. which is not used in at least two filters.
2. If both INSs suffer a hard failure at any point in time, the hard failure isolation procedures will flag all sensors as suffering hard failures — hard failure isolation becomes ambiguous. Simultaneous failure of any other two sensors (one INS and one aiding sensor or two aids) will normally produce proper isolation.

Selection of Hard Failure Isolation Filters

Hard failure isolation is initiated by summing the MeasFailed and MeasFormed counters from all hard failure isolation (HFI) filters. The selection of these filters depends on the current status of all sensors. Let's list some basic concepts to act as framework for selecting the most appropriate filters for hard failure isolation:

1. The two filters judged to be most effective for HFI of each sensor will be used to isolate a hard failure. Two filters are selected because one filter cannot always reliably isolate a sensor failure; and more than two filters may increase the time to isolation.
2. The hard failure of any aiding sensor is best detected using filters which use the failed sensor as well as the most accurate healthy sensor.
3. Where possible, do not select any filter for the hard failure isolation of its most accurate aiding sensor. There are two reasons for this:
 - a. The error states used to model the behaviour of the most accurate aiding sensor in any filter will be mostly unobservable. (Note that, even though these states are not very observable, they are still required to maintain a lower bound on the filter's accuracy estimates.) Some of the unobserved systematic errors in the data from that sensor will be forced into the corresponding residuals. If enough of these systematic errors are passed to the residuals, it may cause the residual tests to fail. At best, the filter's most accurate measurements would be improperly rejected. At worst, hard failure false alarms would be signalled.
 - b. Secondly, since the Kalman filter solution will closely track the results of the most accurate aiding sensor, if that sensor suffers a "softer" failure, it will be more difficult to detect via residual testing — the filter will also tend to track the failure. (Note that the soft failure testing will still have a good chance of detecting the failure in the most accurate aid but tends to be slower doing so.) On the other hand, a "softer" failure in an aiding sensor other than the filter's most accurate will be easier to detect. The filter will still track the most accurate aid. The failure will drive the output of the less accurate sensor away from the filter's expectation, allowing the residual tests to identify the divergence.
4. The condition imposed by item 3.a can never be met for the most accurate aid overall. But, the this sensor must still be tested. To minimize the effects listed, special techniques can be employed:
 - a. To avoid excessive false alarms, include any expected systematic errors in the measurement noise model.
 - b. To identify failures as reliably and quickly as possible, use filters which use the second most accurate sensor for HFI of the most accurate sensor.
5. The hard failure of an INS is best detected using the INS1 + INS2 filter and the filter which uses only the failed INS and the most accurate healthy aiding sensor. If the INS1 + INS2 filter cannot be used (because the other INS is not good), select the filter which uses only the most accurate healthy aid and the filter which uses only the second most accurate healthy aid.

Note that the unifiers fill all the requirements for HFI of all aiding sensors.

Hard failure isolation of all non-INS sensors Hard failure isolation of all non-INS sensors proceeds by analyzing each filter as follows.

- I. The filter has NOT suffered a failure of its (soft failure detection) χ^2 test, is NOT the INS1 + INS2 filter, and has a primary INS which is OK for hard failure isolation (OK for HFI — defined as no BIT/DCT status failure and no soft failure):

A. TRUE —

1. The filter uses all "good" aiding sensors (defined as aids used in at least two filters which have NOT suffered a TYPE M hard failure or a soft failure) and only good sensors (These filters will be called the "unifilters" — one for INS1 and one for INS2.):
 - a. TRUE — select this filter for hard failure isolation of all good aiding sensors. (If both INSs are OK for HFI, both unifilters will be selected. If only one INS is all OK, only the good unfilter will be selected. The second filter is selected below.)
 - b. FALSE — The primary INS for this filter is OK for HFI but the other INS is not, and all good sensors except one are used in this filter (This is one of the non-unfilter options mentioned above — if only one INS is OK for HFI, both filters must be selected from the good INS's branch of filters. The reliability of the hard failure isolation is significantly reduced because the two filters selected will be very similar.):
 - i. TRUE — The least accurate good aid is NOT used in this filter:
 - ☐ TRUE — select this filter for hard failure isolation of all aiding sensors except the least accurate good aid. (This option will result in selection of the filter which is identical to the unfilter selected above except that the least accurate good aid is NOT enabled.)
 - ☐ FALSE — The second least accurate healthy aid is NOT used in this filter (This option will result in selection of the filter which is identical to the unfilter selected above except that the second least accurate good aid is NOT enabled.):
 - TRUE — select this filter for hard failure isolation of the least accurate good aid.
 - FALSE — do NOT select this filter.
 - ii. FALSE (both INSs are OK for HFI BUT not all good aids are in this filter) — This option will be selected when one unfilter has suffered a χ^2 failure (making that unfilter ineligible for HFI) but a sensor soft fail has not been declared. For each good aid in the filter, proceed as follows. This filter uses the aid being tested AND the largest number of additional aids:
 - TRUE — select this filter for hard failure isolation of the aid being tested.
 - FALSE — do NOT select this filter for this aid.

- B. FALSE (this filter has failed its χ^2 test OR is the INS1 + INS2 filter OR has a primary INS which is NOT OK for HFI) — do NOT select this filter.

Hard failure isolation of INSs Hard failure isolation of any INS proceeds by analyzing each filter as follows.

- I. This filter has NOT suffered a failure of its soft failure (χ^2) test:
 - A. TRUE — The INS1 + INS2 filter can be used for hard failure isolation (both INSs are OK for HFI AND the INS1 + INS2 filter has NOT suffered a χ^2 test failure):
 1. TRUE —
 - a. This filter is the INS1 + INS2 filter:
 - i. TRUE — select this filter for hard failure isolation of both INSs.
 - ii. FALSE — The filter uses only the most accurate good aid (defined as the most accurate aid which is used in at least two filters and has NOT suffered a TYPE M hard failure or a soft failure):
 - TRUE — select this filter for hard failure isolation of its primary INS. (There should be one of these filters for HFI of each INS.)
 - FALSE — do NOT select this filter.
 2. FALSE (The INS1 + INS2 filter CANNOT be used for HFI.) — Only one aid is enabled in this filter:
 - i. TRUE — The enabled aid is the most accurate good aid OR the second most accurate good aid:
 - TRUE — select this filter for hard failure isolation of its primary INS.
 - FALSE — do NOT select this filter.
 - ii. FALSE — do NOT select this filter.
 - B. FALSE — do NOT select this filter (it has failed its χ^2 test).

Hard Failure Isolation Examples To illustrate the hard failure isolation procedures just described, a two INS, two aiding sensor configuration will be used. The standard three aid DIINS configuration adds significantly to the complexity of the illustrations but does contribute much to the understanding of the concepts. The INSs are assumed to be identical; Aid1 is assumed to be more accurate than Aid2. The standard DIINS filter configuration with this sensor suite is presented in terms of sensors in Figure 2.1 and in terms of filter numbers in Figure 2.2.

Figure 2.1 Standard DIINS Filter Configuration in Terms of Sensors

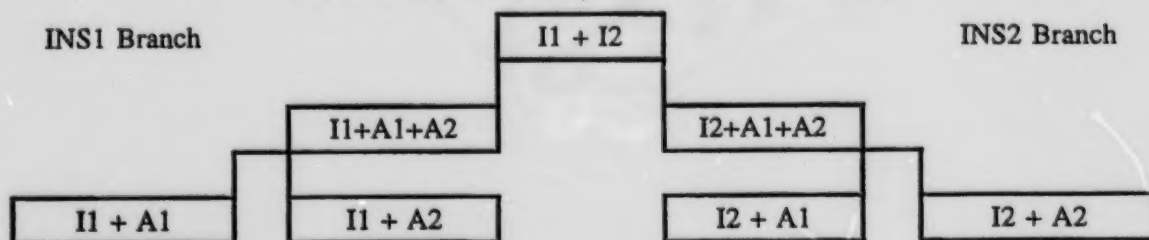
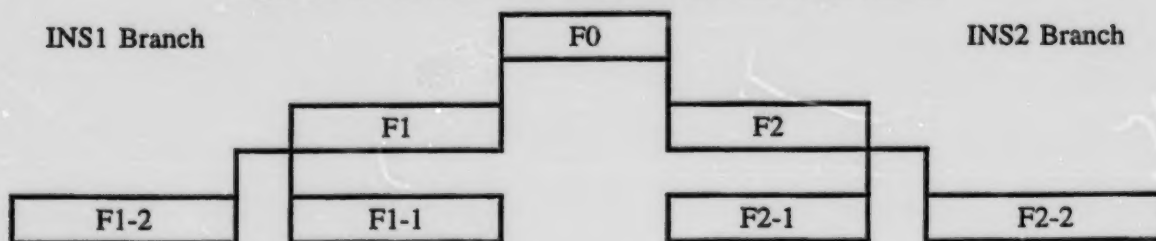


Figure 2.2 Standard DIINS Filter Configuration in Terms of Filter Numbers



where

1. F0 is the INS1 + INS2 filter;
2. F1 and F2 are the INS1 and INS2 unifiers (with Aid1 and Aid2 measurements), respectively;
3. FI-J is the filter based on INS(I) and using all aiding sensors except J.

Table 2.2 illustrates the isolation of hard sensor failures when there are no χ^2 test failures and no soft sensor failures. Note that this table is used under all other failure conditions. No filter which has suffered a χ^2 test failure will be used for hard failure isolation.

Table 2.2 "No Fail" Hard Sensor Failure Isolation

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
0	X ₂	0	X ₁	0	0	0	INS1
0	0	0	X ₁	0	X ₂	0	INS2
0	0	X ₁	0	X ₂	0	0	Aid1
0	0	X ₁	0	X ₂	0	0	Aid2

where

1. X indicates that the MeasFormed and MeasFailed counters for that filter should be included in the TotalMeas counters (the subscript refers to the array address in the software);
2. 0 indicates that the results are not used in the isolation procedure.

What about HFI after a soft sensor failure?

1. After one INS has suffered a soft failure, HFI for the failed INS is suspended. HFI for the three remaining sensors uses the three filters which do not use the failed INS.

Table 2.3 Hard Sensor Failure Isolation After an INS1 Soft Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
—	—	—	—	0	0	0	INS1
—	—	—	—	0	X ₁	X ₂	INS2

Table 2.3 (Continued) Hard Sensor Failure Isolation After an INS1 Soft Failure

—	—	—	—	X ₁	X ₂	0	Aid1
—	—	—	—	X ₁	0	X ₂	Aid2

Table 2.4 Hard Sensor Failure Isolation After an INS2 Soft Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
X ₂	X ₁	0	—	—	—	—	INS1
0	0	0	—	—	—	—	INS2
0	X ₂	X ₁	—	—	—	—	Aid1
X ₂	0	X ₁	—	—	—	—	Aid2

where — indicates that the filter is excluded from HFI because of possible corruption (in this case, as a result of the primary INS soft failure).

If another sensor fails after the first INS, no subsequent failure isolation is possible.

- a. If the other INS subsequently fails, there will be no filter with a healthy primary INS, and thus no filter will be eligible for hard failure isolation.
 - b. If either aiding sensor fails after the INS has failed, there will only be one filter which does not use a failed sensor. For example, if Aid1 failed after INS1 had failed, F2-1 would be the only failure free filter. Hard failure detection is still possible using F2-1, but there is no redundancy left to allow further hard sensor failure isolation.
2. After an aiding sensor has suffered a soft failure, HFI for the failed aid is suspended. Assuming that one aid has suffered a soft failure, HFI for the three remaining sensors proceeds according to following tables.

Table 2.5 Hard Sensor Failure Isolation After an Aid1 Soft Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
X ₂	—	—	X ₁	—	—	0	INS1
0	—	—	X ₁	—	—	X ₂	INS2
0	—	—	0	—	—	0	Aid1
X ₁	—	—	0	—	—	X ₂	Aid2

Table 2.6 Hard Sensor Failure Isolation After an Aid2 Soft Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
—	X ₂	—	X ₁	—	0	—	INS1
—	0	—	X ₁	—	X ₂	—	INS2

Table 2.6 (Continued) Hard Sensor Failure Isolation After an Aid2 Soft Failure

—	X_1	—	0	—	X_2	—	Aid1
—	0	—	0	—	0	—	Aid2

If another sensor fails after the first aid, no subsequent failure isolation is possible.

- a. If either INS was the next to suffer a soft failure, there would only be one filter which does not use a failed sensor. For example, if INS1 failed after Aid1, F2-1 would still be failure free. This is like 1.b.
 - b. If the second aid suffered a soft failure after the first, only filter F0 would be failure free. All other filters will be propagating without measurements meaning that accuracy and reliability will decrease with time.
3. A special case occurs when the INS1 + INS2 filter suffers a χ^2 test failure. If no sensor failures have been isolated: If the INS1 + INS2 filter suffers a χ^2 test failure and an INS

Table 2.7 Hard Sensor Failure Isolation After a χ^2 Failure of the INS1 + INS2 Filter

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Hard Failure
X_2	X_1	0	—	0	0	0	INS1
0	0	0	—	0	X_1	X_2	INS2
0	0	X_1	—	X_2	0	0	Aid1
0	0	X_1	—	X_2	0	0	Aid2

has suffered a soft failure, HFI will proceed just as it did in Tables 2.3 and 2.4. Notice that the INS1 + INS2 filter has already been excluded in these tables. If the INS1 + INS2 filter suffers a χ^2 test failure and an aiding sensor has suffered a soft failure, HFI for the unfailed aid will proceed as it did in Tables 2.5 and 2.6. However, HFI for the INSs will not be possible under this scenario.

In the Introduction, it was suggested that a soft failure could be most accurately defined as "the result of unmodelled errors which are large enough to corrupt filtered results but cannot be identified by hard failure detection techniques". This pragmatic definition is a useful place to begin a discussion of soft failure detection.

This definition states that filter results will be corrupted by the effects of a soft sensor failure, ie. filter performance may be worse than expected. Further, it states that the effects of a soft failure cannot be detected using "hard failure detection techniques", ie. residual tests. Assuming that errors capable of causing soft failures are **possible**, it is important that additional failure detection schemes be devised to detect soft sensor failures. The specific scheme developed for DIINS is described below.

3.1 Soft Failure Detection

Various procedures which can be used for soft failure detection are described in the literature. The procedures used in DIINS were presented in reference [5]. In brief, the test is based on a comparison between two Kalman filter solutions: one is the usual filter with regular measurement updates from independent sensors; the second solution uses the same initial conditions and propagation models but does not perform any measurement updates. In *ibid.*, a statistical test is developed which looks for unexpected differences in the two solutions. A statistically significant difference triggers an assumption that the filter has experienced the soft failure of **any one** of its component sensors. The test is known as the chi-squared (χ^2) test because the χ^2 distribution is used to test the computed statistic. The basis for the test is presented below.

The Theoretical χ^2 Soft Failure Detection Test

The Kalman filter state propagation model from time t_{k-1} to time t_k can be represented by the following recursive equation

$$\vec{x}_k = \Phi_{k,k-1} \vec{x}_{k-1} + \vec{w}_{k-1} \quad (3.1)$$

The linear measurement model at time t_k is

$$\vec{z}_k = H_k \vec{x}_k + \vec{v}_k \quad (3.2)$$

Vectors $\vec{w}$ and $\vec{v}$ are assumed to be independent (of one another), zero mean, Gaussian white sequences with covariance matrices Q and R , respectively. The initial state vector $\vec{x}_0$ is assumed to be a Gaussian random vector independent of $\vec{w}$ and $\vec{v}$ with covariance P_0 .

State estimates are computed using the following equations. Both filters are initialized to

$$\begin{aligned} \hat{\vec{x}}(t_0) &= E[\vec{x}(t_0)] = \vec{x}_0 \\ P(t_0) &= P_0 \end{aligned} \quad (3.3)$$

After propagation from time t_{k-1} to time t_k ,

$$\begin{aligned}\hat{\tilde{x}}_k(-) &= \Phi_{k,k-1} \hat{\tilde{x}}_{k-1} \\ P_k(-) &= \Phi_{k,k-1} P_{k-1} \Phi_{k,k-1}^T + Q_{k-1}\end{aligned}\quad (3.4)$$

This is the only step used for the no measurement filter. If an independent measurement is available (and used) at time t_k , and if the measurement can be expressed as a linear combination of states, the second filter is updated using

$$\begin{aligned}\hat{\tilde{x}}_k(+) &= \hat{\tilde{x}}_k(-) + K_k (\bar{z}_k - H_k \hat{\tilde{x}}_k(-)) \\ P_k(+) &= (I - K_k H_k) P_k(-)\end{aligned}\quad (3.5)$$

This is the step that differentiates the two filters.

To describe the χ^2 test, begin by defining the errors in the state estimates:

$$\tilde{\tilde{x}} = \tilde{x} - \hat{\tilde{x}} \quad (3.6)$$

Since $\hat{\tilde{x}}$ is assumed Gaussian, $\tilde{\tilde{x}}$ can be assumed to be Gaussian with a zero mean (or a standard normal distribution). At this point, all variables are assumed to refer to the same point in time, and the subscripts 1 and 2 will be used to indicate the estimates with and without measurements, respectively.

The difference in the errors of the two state estimates is now defined as

$$\begin{aligned}\vec{\beta} &= \tilde{\tilde{x}}_1 - \tilde{\tilde{x}}_2 \\ &= (\tilde{x} - \hat{\tilde{x}}_1) - (\tilde{x} - \hat{\tilde{x}}_2) \\ &= \hat{\tilde{x}}_2 - \hat{\tilde{x}}_1\end{aligned}\quad (3.7)$$

Since both state error vectors are assumed to be standard normal, $\vec{\beta}$ can be assumed to be standard normal. The covariance matrix of $\vec{\beta}$ is

$$\begin{aligned}B &= E[\vec{\beta} \vec{\beta}^T] \\ &= E[\tilde{\tilde{x}}_1 \tilde{\tilde{x}}_1^T - \tilde{\tilde{x}}_1 \tilde{\tilde{x}}_2^T - \tilde{\tilde{x}}_2 \tilde{\tilde{x}}_1^T + \tilde{\tilde{x}}_2 \tilde{\tilde{x}}_2^T] \\ &= P_1 + P_2 - P_{12} - P_{21}\end{aligned}\quad (3.8)$$

Reference [5] shows that under the conditions of optimal filter gains, identical state models and identical initial conditions for both solutions,

$$P_{21} = P_{12}^T = P_1 \quad (3.9)$$

so that

$$B = P_2 - P_1 \quad (3.10)$$

Further details of this derivation are presented in *ibid.*

The χ^2 (soft failure detection) test is performed on a test statistic which is the (scalar) quadratic form of $\vec{\beta}$:

$$b^2 = \vec{\beta}^T B^{-1} \vec{\beta} \quad (3.11)$$

The quadratic form of a standard normal vector such as $\vec{\beta}$ has a χ^2 distribution with m degrees of freedom, where m is the row dimension of $\vec{\beta}$.

In the absence of soft failures, the probability that b^2 has a χ_m^2 distribution is

$$P\{b^2 \leq \chi_m^2\} = 1 - \alpha \quad (3.12)$$

where $(1 - \alpha)$ is the *confidence level*. Alternatively,

$$P\{b^2 > \chi_m^2\} = \alpha \quad (3.13)$$

where α is the *significance level*. Thus, whenever

$$b^2 > \chi_{m,\alpha}^2 \quad (3.14)$$

a χ^2 failure (ie. soft failure detection) will be declared.

In theory, any value of b^2 has a finite probability. By choosing to signal a failure at the $\chi_{m,\alpha}^2$ point in the distribution, we accept the fact that we risk signalling a failure when none has occurred. This is known as a Type 1 error, and the probability of committing such an error is equal to the significance level, α . Choosing a small value for α implies a reluctance to unjustly signal a failure. However, as the probability of committing a Type 1 error decreases, the probability of committing a Type 2 error increases. A Type 2 error occurs when a failure has occurred but is not detected. Thus, the significance level must be selected to properly balance the probabilities of Type 1 and Type 2 errors. Values of α between 1 and 5 per cent are typically selected.

The DIINS χ^2 Soft Failure Detection Test

In the DIINS implementation of the soft failure detection test, some empirical adjustments have been made to the theory. First of all, only a portion of the state vector is used to compute $\vec{\beta}$. The selection of the optimal subset to be used was made on the basis of empirical knowledge, testing with real data, and the desire to minimize computational requirements. In the discussion below, the state vector is grouped into "system" states and the "sensor error" states, where

1. system states describe the *systemic* errors in the INS' position, velocity and attitude,
2. sensor error states describe errors in individual sensors — INS accelerometer and gyro biases, and aiding sensor errors (eg. GPS position or Loran-C TDs).

Sensor error states are all modelled as first-order Gauss-Markov processes. In the absence of measurements, the sensor error states will drive the changes in the system states.

In brief,

1. Simulated soft INS failures (eg. an increase in the bias of one accelerometer) demonstrated that the b^2 almost always climbed well above the test limit after the failure. The time between initiation of the failure and detection varied with the states selected.
2. In testing of real sea trial data, soft failure detection was not necessarily so straight forward: often soft failures were identified then quickly recovered, or soft failures were detected using some state subsets but not with others. In the real world, there are many unexpected circumstances which can cause sensors to deviate from their expected behaviour. These deviations are not necessarily failures but may be identified as such by χ^2 testing.
3. Using the full state vector required too much computational effort and seemed to cause the soft failure detection to be prone to Type 2 errors: using too many states may cause the effects of a failure to be "diluted" too much to be detected.
4. Excluding all system states (ie. using some or all sensor error states) seemed to result in a tendency to Type 1 errors: unmodelled behaviour may be interpreted as a failure.
5. Using all the system states and only the system states seems to produce a good balance between Type 1 and Type 2 errors:
 - a. Only those components of the state vector which are of interest to the user (ie. the navigation components — position, velocity and attitude) are included.
 - b. The sensor error states are propagated into the system states. Thus, a sensor failure which may be first reflected in the sensor error states will (after some delay) be observable in the system states. This may help alleviate Type 1 errors by "smoothing" out transient, unmodelled effects.
 - c. Attitude errors propagate into velocity errors and velocity errors propagate into position errors. Including position states from the soft failure detection test seems to produce an decrease in Type 1 errors by smoothing out transient effects present in the velocity and attitude states. Excluding velocity and attitude states seems to produce an increase in Type 2 errors: too much smoothing delays or prevents soft failure detection.
6. The effects of selecting a specific subset of states can be moderated by selection of a complementary significance level: the probabilities of Type 1 and Type 2 errors are directly affected by the choice of α .

A further comment on implementation of the χ^2 soft failure detection test: note that at any time after the first measurement has been processed by Filter 1,

$$\begin{aligned} \hat{x}_1 &\neq \hat{x}_2 \\ P_1 &< P_2 \end{aligned} \tag{3.15}$$

Also, covariance matrix, B , will be positive definite unless $P_1 = P_2$, in which case, $B = 0$ and B^{-1} is undefined. In practice, even after processing a few measurements, $P_1 \simeq P_2$, and B will be numerically unstable. In DIINS, B is tested for numerical stability by computing a condition number. If the condition number indicates an unstable matrix, no failure detection is performed.

Instead, a message stating that "... ChiSquared changed to Status = BAD_COND ..." is sent to the operator.

When the χ^2 soft failure detection test changes from the failed condition to the passed condition, a message of the form "... ChiSquared changed to Status = OK ..." is sent to the operator. If the status changes passed to failed, "OK" is replaced by "FAILED". The status is initialized to "OK".

Changes in χ^2 test status are NOT sent to the FDI processor directly. Instead,

1. If the status changes and remains in the changed state continuously for a predetermined length of time, and
2. If the status is NOT changing to BAD_COND (poor conditioning of the B-matrix),

then a χ^2 failure flag is sent to the FDI processor. This delay prevents spurious or transient effects from affecting FDI results. A change in the FDI flag will result in a message of the form "... ChiSquared changed to Failure = TRUE (or FALSE) ..." being sent to the operator.

3.2 Soft Failure Isolation

Soft failure isolation (SFI) is very similar to hard failure isolation:

1. Soft failure detection results from more than one filter are required.
2. For each sensor, a group of *soft failure isolation filters* is selected.
3. A soft sensor failure isolation is declared only when all SFI filters signal the failure.
4. Recovery from a soft sensor failure is declared as soon as any SFI filter recovers.

However, whereas hard failure detection always identifies two suspect sensors, soft failure detection identifies at least two but possibly more than two, depending on the number of aiding sensors in a particular filter. In addition, there is only one soft sensor failure type instead of the two found in hard failure isolation. Therefore, in general, the filters selected for hard failure isolation will NOT be the best filters for soft failure isolation.

Specifically, to isolate soft sensor failures, at each FDI epoch the χ^2 failure flags from all filters are sent to the FDI processor. Then, for each sensor (K), if a soft failure has been detected in every sensor K SFI filter, a sensor K soft failure isolation is declared:

1. the 'SOFT_FAILURE' sensor status flag is set to TRUE,
2. a message of the form "FDI: Sensor K SOFT failure ISOLATION ..." is sent to the operator.

If the χ^2 failure flag is FALSE in even one SFI filter, it is assumed that the sensor has not suffered (or has recovered from) a soft failure.

A sensor which has previously been identified as having suffered a soft failure can recover to the healthy state. This occurs when one or more of the SFI filters for that sensor sends a χ^2 failure flag which is FALSE. If sensor K recovers from a soft failure, a message of the form "FDI: Sensor K SOFT failure RECOVERY ..." is sent to the operator.

Again, examples will be used to illustrate the soft failure isolation process.

1. Assume that a gyro in an INS starts performing below its expected level.
 - a. After some time, χ^2 tests in filters using that INS will start to fail.

- b. Each filter will start its timer when the χ^2 test first fails.
- c. As soon as the timer reaches its predefined limit, each filter will notify the central FDIR processor.
- d. When all failing INS' SFI filters send χ^2 failure flags which are TRUE, the FDIR processor will declare an INS soft failure.
- e. The FDIR processor notifies the operator who can attempt to locate the source of the soft failure isolation (ie. the substandard gyro).
- f. If the source is located and the problem fixed (eg. by replacing the bad gyro), the failure isolation process can be reversed:
 - the χ^2 tests will start passing;
 - the good status timer starts;
 - after a predefined delay, the FDIR processor is notified of the recovery (by sending it χ^2 failure flags which are set to FALSE);
 - the FDIR processor identifies the INS recovery and notifies the operator.

Similar to hard failure isolation,

1. Soft failure isolation will NOT be declared for a sensor
 - a. which is currently suffering from a hard failure;
 - b. which is not used in at least two filters.
2. If both INSs suffer a soft failure at any point in time, the soft failure isolation procedures will flag all sensors as suffering soft failures, and soft failure isolation becomes ambiguous. Simultaneous failure of any other two sensors (one INS and one aiding sensor or two aids) will normally produce proper isolation.

Selection of Soft Failure Isolation Filters

The selection of these filters depends on the current status of all sensors. Let's list some basic concepts to act as a framework for selecting the most appropriate filters for soft sensor failure isolation:

1. More than one filter is always required to declare a soft sensor failure.
2. Soft failure isolation is more reliable when filters using the minimum number of sensors (one INS and one aid, if possible) are used.
3. The soft failure of an INS is best detected with the INS1 + INS2 filter as well as the filter which uses only the failed INS and the most accurate healthy (non-INS) aiding sensor.
4. The soft failure of any aiding sensor is best detected using filters which use the failed sensor and the minimum number of additional aids (none if possible). If only one aiding sensor is activated, a χ^2 failure must be attributed to one of only two sensors, the primary INS or the aid. In a filter using additional aiding sensors, a χ^2 failure may be attributed to any of the additional aids. This dilution tends to reduce the reliability of the soft failure isolation process.
5. When only one INS is OK for SFI, both filters must be selected from the good INS's branch of the filter tree. The filter using only the good INS and the aid being tested can still be selected. However, the second filter must use more than one aid: it must include the tested

aid and at least one other. If a filter using an aiding sensor which is more accurate than the tested aid is selected, the more accurate sensor may mask a soft failure of the tested sensor (due to the structure of the DIINS χ^2 test — ie. the use of system states only). The chance of a Type 2 error (missing a soft failure) is increased. A filter using only the tested aid and a less accurate aid will be similar to the first filter selection (it will be dominated by the tested aid), and the χ^2 test results will likely be very similar. The chance of a Type 1 error (signalling a soft failure when none has occurred) is increased. In either case, the selection process becomes less reliable and less straight forward.

A compromise between increased probabilities for Type 1 and Type 2 errors has been selected: the filter which has the tested aid as its most accurate aid and which uses ALL aids which are less accurate than the tested aid will be used when possible. This choice, in general, will increase the probability of a Type 1 error: better to err on the side of safety. If the tested aid is the least accurate of all those being used, the filter using only the tested aid and the one healthy aid which is the next most accurate will be used. For this sensor, an increased risk of a Type 2 error is unavoidable, but selecting the two least accurate aids should minimize the risk.

Soft failure isolation of all non-INS sensors Soft failure isolation of all non-INS sensors proceeds by analyzing each filter as follows.

- I. This is NOT the INS1 + INS2 filter, AND the primary INS is OK for soft failure isolation ("OK for SFI" — defined as no BIT/DCT status failure and no TYPE M or A hard failure):
 - A. TRUE — There is only one aiding sensor activated in this filter:
 1. TRUE — The activated sensor has NOT suffered a TYPE M or A hard failure:
 - a. TRUE — select this filter for soft failure isolation of that aid. (If both INSs are OK for SFI, there will be one filter selected which uses INS1 and another which uses INS2.)
 - b. FALSE — do NOT select this filter for SFI of any aid.
 2. FALSE (more than one aid active) — The primary INS for this filter is OK for SFI but the other INS is NOT (both SFI filters will have to be selected from this branch of filters):
 - a. TRUE —
 - i. Only the two least accurate "good" aiding sensors (ie. aids with NO hard failure) are active in this filter:
 - TRUE — select this filter for SFI of the least accurate good aid.
 - FALSE — do NOT select this filter for SFI of the least accurate good aid.
 - ii. The most accurate aiding sensor in this filter has NOT suffered any hard failure, and this filter uses ALL healthy aids which are less accurate than the tested aid:
 - TRUE — select this filter for SFI of its most accurate aid.
 - FALSE — do NOT select this filter for SFI of its most accurate aid.

- b. FALSE (both INSs OK for SFI and more than one aid active) — do NOT select this filter for SFI of any aid.

- B. FALSE (this is the INS1 + INS2 filter, or the primary INS is NOT OK for SFI) — do NOT select this filter for SFI of any aid.

Soft failure isolation of INSs Soft failure isolation of any INS proceeds by analyzing each filter as follows.

- I. Both INSs are "OK for soft failure isolation" (OK for SFI — defined as no BIT/DCT status failure and no TYPE M or A hard failure):

- A. TRUE —

- 1. This filter is the INS1 + INS2 filter:

- a. TRUE — select this filter for soft failure isolation of both INSs.
- b. FALSE — The filter uses only the most accurate healthy aid (defined as having NO hard failure or soft failure):
 - i. TRUE — select this filter for soft failure isolation of its primary INS.
 - ii. FALSE — do NOT select this filter for SFI of any INS.

- B. FALSE — The primary INS for this filter is OK for SFI (but the other INS is not):

- 1. TRUE — Either the most accurate healthy aid OR the second most accurate healthy aid is the only aid active in this filter:
 - a. TRUE — select this filter for soft failure isolation of its primary INS.
 - b. FALSE — do NOT select this filter for SFI of either INS.
- 2. FALSE — do NOT select this filter for SFI of either INS.

Soft Failure Isolation Examples The two INS, two aiding sensor configuration described in 3 is again used here to illustrate the soft failure isolation procedures. This configuration is presented graphically in Figures 2.1 and 2.2.

Table 3.1 illustrates the isolation of soft sensor failures when there are no TYPE M or A hard sensor failures. Note that this table is used under all other failure conditions.

Table 3.1 "No Fail" Soft Sensor Failure Isolation

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Soft Failure
0	X ₂	0	X ₁	0	0	0	INS1
0	0	0	X ₁	0	X ₂	0	INS2
0	X ₁	0	0	0	X ₂	0	Aid1
X ₁	0	0	0	0	0	X ₂	Aid2

where

- 1. F0, F1 and FI-J were defined for hard failure isolation.

2. X indicates that the filter failed the χ^2 test (the subscript refers to the array address in the software);
3. 0 indicates that the results are not relevant to the isolation procedure.

What about SFI after a hard sensor failure?

1. After one INS has suffered a TYPE M or A hard failure or a BIT/DCT status failure, SFI for the failed INS is suspended. SFI for the three remaining sensors uses the three filters which do not use the failed INS.

Table 3.2 Soft Sensor Failure Isolation After an INS1 Hard Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Soft Failure
—	—	—	—	0	0	0	INS1
—	—	—	—	0	X ₁	X ₂	INS2
—	—	—	—	X ₁	X ₂	0	Aid1
—	—	—	—	X ₁	0	X ₂	Aid2

Table 3.3 Soft Sensor Failure Isolation After an INS2 Hard Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Soft Failure
X ₂	X ₁	0	—	—	—	—	INS1
0	0	0	—	—	—	—	INS2
0	X ₂	X ₁	—	—	—	—	Aid1
X ₂	0	X ₁	—	—	—	—	Aid2

where — indicates that the filter is excluded from SFI because measurements using the tested sensor have been suspended as a result of the stated failure. The χ^2 test is ineffective in these instances.

If another sensor fails after the first INS, no subsequent failure isolation is possible.

- a. If the other INS subsequently fails, there will be no filter with a healthy primary INS, and thus no measurements being formed in any filter.
- b. If either aiding sensor fails after the INS has failed, there will only be one filter which does not use a failed sensor. For example, if Aid1 failed after INS1 had failed, F2-1 would be the only failure free filter. Soft failure detection is still possible using F2-1, but there is no redundancy left to allow further soft sensor failure isolation.

2. After an aiding sensor has suffered any hard failure, SFI for the failed aid is suspended. Assuming that just one aid has suffered a hard failure, SFI for the three remaining sensors proceeds according to following tables.

Table 3.4 Soft Sensor Failure Isolation After an Aid1 Hard Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Soft Failure
X ₂	—	—	X ₁	—	—	0	INS1
0	—	—	X ₁	—	—	X ₂	INS2
0	—	—	0	—	—	0	Aid1
X ₁	—	—	0	—	—	X ₂	Aid2

Table 3.5 Soft Sensor Failure Isolation After an Aid2 Hard Failure

F1-1	F1-2	F1	F0	F2	F2-2	F2-1	Soft Failure
—	X ₂	—	X ₁	—	0	—	INS1
—	0	—	X ₁	—	X ₂	—	INS2
—	X ₁	—	0	—	X ₂	—	Aid1
—	0	—	0	—	0	—	Aid2

If another sensor fails after the first aid, no subsequent failure isolation is possible.

- a. If either INS was the next to suffer a (status or hard) failure, there would only be one filter not using a failed sensor. For example, if INS1 failed after Aid1, F2-1 would still be failure free. This is like 1.b.
- b. If the second aid suffered a (hard) failure after the first, only filter F0 will be failure free. All other filters would be propagating without measurements.

Chapter 4

SELECTION OF OPTIMAL SOURCE FOR OUTPUT

In earlier chapters, we looked at procedures for failure detection (in the form of data quality and residual tests for "hard" failures and χ^2 tests for "soft" failures) and failure isolation (in the form of logic tables based on the failure detection results). In and of themselves, these tests have limited value. To be really useful, the information from these tests must be used to

1. identify unreliable filters,
2. protect uncorrupted filters from the effects of sensor failures,
3. select the "best" source of navigation output.

Filter reliability is quantified through analysis of χ^2 test results and the health of each component sensor. Sensor health is defined as

1. Good BIT/DCT status,
2. And no hard or soft failures.

Filters can be protected from sensor failure corruption in two ways:

1. By disabling measurements (this technique is called *measurement reconfiguration* and is fully discussed in Section 4.2);
2. By stopping, reconfiguring and restarting filters in response to changes in sensor health (this technique is called *filter reconfiguration* and is briefly described below);

A technique called *filter classification* ranks filters according to the sensor and filter status. The highest ranked filter is selected for output. This technique is used in conjunction with measurement reconfiguration, but is an alternative to filter reconfiguration. It is described in Section 4.1. The concepts used in filter reconfiguration and classification are similar, but the implementation is quite different. Basic ideas are described in reference [3].

Selection of the best source for output must be based on a combination of reliability and accuracy. Failure detection and isolation are used to maximize reliability. Knowledge of the accuracy expected from each filter and sensor can be used to judge accuracy. The benefits and risks associated with reliability must be weighed against those of accuracy to arrive at the optimum choice for navigation output. In some cases, maximum reliability and maximum accuracy coincide. In other cases, increased reliability will lead to decreased accuracy or vice versa. This is where the problem gets interesting.

Filter reconfiguration and classification both rely on a specific filter tree structure: the INS1 + INS2 filter forms the "root"; then there are two main branches which are mirror images of one another, one for INS1 filters and one for INS2 filters. The top level of each branch contains the so-called *unifilter* which uses all available aiding sensors. The second level contains a number of filters, each of which excludes one aiding sensor. For filter reconfiguration, no further branching is required. However, for filter classification, additional branching may be required (depending on the number of aiding sensors used). Because filter classification does not "reconfigure" any filter, all required combinations of sensors must be initialized. The third level would contain filters which each exclude two aiding sensors. This process continues until there is a level

consisting of filters with one aiding sensor each. This guarantees that there will be a filter which cannot be corrupted by a sensor failure (provided there is at least one healthy INS and one healthy aiding sensor).

Filter classification has the following advantages:

1. It is easier to implement.
2. Historical information stored within the filters' recursive formulation is preserved (although it is potentially corrupt).

And disadvantages

1. A filter which has been excluded from consideration for output because of BIT/DCT sensor status failure, hard or soft sensor failures, or χ^2 test failure is never reconfigured. This increases the risk of using a corrupt filter for output. To deal with this increased risk, an excluded filter can be treated in one of two ways:
 - a. It may be permanently excluded. This maximizes reliability by minimizing the possibility of using a corrupt filter. It also reduces possible accuracy and presents the likelihood that output accuracy may decrease as mission length increases (the probability that the most accurate filters will be excluded increases with time).
 - b. It may be included in consideration for output after the failures condition(s) has been removed. This increases the risk of a corrupt filter being used for output, thus decreasing reliability. At the same time, potential output accuracy is increased — the historical information stored in the filters can increase accuracy if it is reliable but is potentially dangerous if it is corrupted. Levels of corruption are very difficult to quantify.
2. As the number of available sensors grows, the number of required filters grows exponentially.

Filter reconfiguration has some advantages:

1. When sensors fail and then recover, a filter using all healthy sensors can be used for output with no risk that it has been corrupted.
2. Only two levels of filters are required, regardless of the number of sensors (each added sensor requires two additional filters, one in the INS1 branch and one in the INS2 branch). Note that this advantage is questioned in the disadvantages list.

And, disadvantages:

1. Reconfiguration is more complicated to implement, requiring inter-filter communication, etc.
2. When a filter is "stopped", all its historical knowledge (eg. INS calibration) is lost.
3. Some FDIR requirements conflict with filter reconfiguration concepts:
 - a. If a sensor failure is isolated, it is desirable to continue to monitor the health of the failed sensor. If it recovers, it can be reintegrated into the FDIR procedures. The filters which signalled the failure should be used to monitor the failed sensor. If those filters are reconfigured without the failed sensor, sensor recovery procedures will be precluded. If those filters are not reconfigured, new filters will have to be introduced into the FDIR procedures. It is difficult to start new filter software tasks on the fly. Initializing all

filters with all possible sensor combinations removes most of the advantages of the filter reconfiguration scheme.

- b. Single aid filters were recommended for both hard and soft failure isolation procedures. If there are 3 or more aiding sensors, filter reconfiguration does not require single aid filters. Again, if the single aid filters are initiated for sensor failure isolation, some of the advantage of filter reconfiguration is eliminated. Note that if there are more than 3 aids, it should be possible to eliminate at least one level of filters — those between the second level (each excluding one aid) and the single aid filters.

At present, DIINS utilizes filter classification techniques for FDIR. These are explained in more detail Section 4.1. Filter reconfiguration will be discussed no further in this document.

Filter classification is based on the following **Basic Principles**:

1. The INS1 + INS2 filter will never be used for output.
2. Any other filter will be considered for output only if its primary INS is perfectly healthy (no BIT/DCT status, hard or soft failure).
3. A filter suffering from a χ^2 test failure will never be used for output.
4. It is assumed that a filter will not be unacceptably corrupted by the hard failure of any aiding sensor.
5. It is assumed that a filter WILL be unacceptably corrupted if the Most Accurate Available Aid (MAAA) suffers a soft failure but NOT when any Other Available Aid (OAA) suffers a soft failure.

In other words, a filter will be considered for output only if

1. it is NOT the INS1 + INS2 filter;
2. and, its primary INS has good Status (eg. no BIT/DCT status failure) and has not suffered any soft or hard failure;
3. and, it has not suffered a χ^2 test failure;
4. and, its most accurate aiding sensor has not suffered a soft failure;
5. and, it has at least one healthy (non-INS) aiding sensor.

On the contrary, a filter will **not** be considered for output if

1. it is the INS1 + INS2 filter;
2. or, its primary INS has bad Status or has suffered any hard or a soft failure;
3. or, the filter has suffered a χ^2 test failure;
4. or, its most accurate aiding sensor has suffered a soft failure.
5. or, it does not have at least one healthy (non-INS) aiding sensor.

4.1 Filter Classification

At present, DIINS uses filter classification: all filters are started at initialization and continue to run until turnoff regardless of failure status. The best filter for output is selected on the basis of sensor health, expected filter accuracy, etc. Details of the DIINS filter classification process follow.

In filter classification, if any filter or sensor recovers from any failure condition, it is assumed that any residual effects arising from the failure condition are negligible. This implies that a recovery should not be signalled until all filters affected by the failure have been sufficiently recalibrated to remove the effects of possible corruption. In the case of recovery from bad INS Status or any hard failure, Basic Principle 4 above states that we can assume no filter corruption. And, the χ^2 test used for soft failure detection, has a kind of built in corruption sensing capability. Time delays in changing these conditions are implemented to avoid excessive switching during transitions in health conditions.

Selection of the Best Filter for Output

To begin the process of finding the best output source, first identify all filters which satisfy all Basic Principles. For each filter, perform the following set of tests.

- A. This is NOT the INS1 + INS2 filter, AND this filter has NOT suffered a χ^2 test failure, AND the primary INS for this filter has NOT suffered a (BAD Status OR hard failure OR soft failure):
 1. TRUE — This filter uses NO sensor which has suffered a soft failure:
 - a. TRUE — consider this filter for output.
 - b. FALSE — There is at least one healthy aiding sensor which is more accurate than the aid which has suffered the soft failure:
 - TRUE — consider this filter for output.
 - FALSE — do NOT consider this filter for output.
 2. FALSE — do NOT consider this filter for output.

Each filter eligible for consideration as a candidate for DIINS output is assessed to identify the one which is "best".

1. More than one eligible filter was identified:
 - a. TRUE — Test each eligible filter as follows. This filter has the lowest filter predicted radial position error (highest expected accuracy) of all eligible filters:
 - TRUE — select this filter for DIINS output. (In practice, all filters whose radial position errors are not significantly larger than the most accurate are considered. This list is reduced by eliminating filters which do not use the most accurate healthy aid. Then filters which have suffered any sensor failure are eliminated. The remaining filter(s) which use the largest number of aids are identified. If one of these filters is currently used for output, continue to use it. If the current output filter is no longer eligible, use the most accurate of those remaining.)
 - FALSE (not the most accurate) — do NOT select this filter for DIINS output.
 - b. FALSE — There was exactly ONE eligible filter identified:
 - TRUE — select that filter for DIINS output.
 - FALSE (NO eligible filter was identified) — find the best SENSOR for output (see Topic 2).

Selection of the Best Sensor for Output

If there is no filter eligible for output, try to find the best sensor. Each active sensor is tested as follows:

1. This sensor supplies positions, or positions can be computed using sensor supplied data:
 - a. TRUE — This sensor has NOT suffered any Hard or Soft (or, for INSS, Status) failure:
 - i. TRUE — This sensor has the highest expected position accuracy of those eligible:
 - TRUE — select this sensor for DIINS output. If necessary, compute positions from available data.
 - FALSE — do NOT select this sensor for DIINS output.
 - ii. FALSE — do NOT select this sensor for DIINS output.
 - b. FALSE — do NOT select this sensor for DIINS output.

Notify the operator that the DIINS output is coming from the selected sensor and not from a filter.

Sensor output is DIINS' last option. Following the philosophy that questionable output is better than no output, if no sensor was identified above, the criteria are loosened. Each sensor is further tested as follows:

1. This sensor supplies positions, or positions can be computed using sensor supplied data:
 - a. TRUE — There is at least one filter which has NOT failed its χ^2 test and where this sensor has been used to form measurements and at least one measurement has been used to update the filter.
 - i. TRUE — This sensor has the highest expected position accuracy of those eligible:
 - TRUE — select this sensor for DIINS output. If necessary, compute positions from available data.
 - FALSE — do NOT select this sensor for DIINS output.
 - ii. FALSE — do NOT select this sensor for DIINS output.
 - b. FALSE — do NOT select this sensor for DIINS output.

Again, it is important to notify the operator that the DIINS output is coming from the selected sensor. If no sensor meets the relaxed criteria, there will be no valid DIINS output: notify the operator.

The expected position accuracy used to select the best sensor for output may be

1. A generic system estimate (eg. GPS is expected to be more accurate than Loran-C).
2. A sensor supplied estimate.

DIINS uses generic estimates because

1. The generic estimates are simpler (requiring no additional data transfer or computations).
2. In many cases, expected sensor accuracy is assumed to be constant in the Kalman filters.
3. The expected position accuracies are very different for different DIINS sensors. GPS is at least an order of magnitude more accurate than Loran-C which is about an order of magnitude

more accurate than Omega. The INSs will never have expected accuracies better than the best aiding sensor, and the expected INS errors will tend to grow with time between fixes. In steady state, INSs will be selected only if no aiding sensor is available for output. After the failure of a more accurate aid, the INSs may be more accurate than a less accurate aid for some time. Generic estimates should be able to reliably identify the most accurate sensor for output in these conditions.

4. If expected position accuracies are similar for different sensors (eg. P(Y) code GPS vs differential GPS), the choice of any of the similar sensors for output should carry little risk.

Note that there is currently NO capability in DIINS to compute positions from any aiding sensor data: only GPS or INS data (in that order) will be used for output.

How Many Filters?

As mentioned above, as the number of aiding sensors grows, the number of required filters for filter classification grows as well. Assuming there are n aiding sensors, each branch of the DIINS filter tree will have

$$\binom{n}{n} = \frac{n!}{(n)! 0!} = 1 \quad (4.1)$$

unifilters (defined as the filter using all n sensors). Each unifilter branches into

$$\binom{n}{n-1} = \frac{n!}{(n-1)! 1!} = n \quad (4.2)$$

subsidiary filters, each of which excludes one aiding sensor. The next level branches into

$$\binom{n}{n-2} = \frac{n!}{(n-2)! 2!} \quad (4.3)$$

additional filters: all possible combinations of $(n-2)$ aiding sensors. For filter classification, this process must continue until only one aiding sensor is used in each filter:

$$\binom{n}{1} = \frac{n!}{1! (n-1)!} = n \quad (4.4)$$

Thus, the total number of filters required in the complete branch (including the unifilter) is equal to

$$N_B = \sum_{i=n}^1 \frac{n!}{i! (n-i)!} = \sum_{i=1}^n \frac{n!}{i! (n-i)!} \quad (4.5)$$

The total number of parallel filters running simultaneously in DIINS (the INS1 + INS2 filter plus both branches) is $N_T = 2N_B + 1$. For example,

n	$N_B(n)$	$N_T(n)$
1	1	3
2	1+2=3	7
3	1+3+3=7	15
4	1+4+6+4=15	31

By inspection, recursive forms can be written

$$\begin{aligned} N_B(n) &= 2N_B(n-1) + 1 \\ N_T(n) &= 2N_T(n-1) + 1 \end{aligned} \quad (4.6)$$

Also, by inspection,

$$\begin{aligned} N_B(n) &= 2^n - 1 \\ N_T(n) &= 2N_B(n) + 1 = 2^{n+1} - 1 \end{aligned} \quad (4.7)$$

These equations all demonstrate that each added sensor more than doubles (ie. doubles plus 1) the required number of filters for the filter classification scheme.

4.2 Measurement Reconfiguration

The measurement reconfiguration component of FDIR is responsible for protecting filters from the corrupting effects of a failed aiding sensor. Measurement reconfiguration is achieved by disabling specific measurements (ie. preventing particular measurements from updating the state vector).

There are 3 different levels of measurement reconfiguration:

1. a. When a particular residual fails its test, then all of the other measurements in the same Group (eg. GPS positions) which are not independent of the failed measurement are disabled, ie. all dependent types in a Group must pass their residual test before any member will be used to update the filter. This will be called Level 1 reconfiguration. For example, if a GPS latitude measurement fails its residual test, the corresponding longitude and height measurements would be disabled. This level of reconfiguration is limited to a particular Group at a particular epoch.
b. There is no 'memory' in Level 1 reconfiguration: the results of the residual test at time t_i will have no impact on measurement processing at time t_{i+1} .
2. a. When the failure of a measurement Type is detected (according to the criteria described in Chapter 2), that measurement Type will be disabled until a failure recovery is declared. This will be called Level 2 reconfiguration. The Level 2 reconfiguration will cause the Level 1 reconfiguration to disable all other dependent measurements in the same Group. For example, if GPS latitude residuals fail XX% per cent of the time for Y seconds, a GPS latitude failure is declared. As a result, GPS longitude and height measurements are disabled until latitude recovers from the failure.
b. Level 2 reconfiguration is cancelled as soon as the failed measurement Type recovers. As soon as GPS latitude residuals pass XX% of the time for YY seconds, recovery is declared: Level 2 reconfiguration is cancelled; GPS longitude and height measurements are again allowed.

3. Measurement reconfiguration up to this point has occurred in individual filters (ie. at the failure detection level). The next level of reconfiguration occurs after sensor failure isolation in the FDI processor. All measurements in all filters which have been formed using data from a sensor which has suffered a hard failure will be disabled. This will be called Level 3 reconfiguration. (Note that Level 3 measurement reconfiguration is more prominent in Filter Classification than in Filter Reconfiguration.) However, consider the following special situations:

- a. The primary INS is involved in all measurements. Should Level 3 be applied after a primary INS failure? If it were to be applied, no measurements would be processed. Active measurements may be able to track unmodelled errors to some degree (depending on the size and rate of change of the errors). However, if all measurements are disabled, the filter will be forced to propagate INS error states which have been calibrated with good measurements but may be changing unpredictably. A strategy of continued measurement formation after a primary INS failure produces benefits but no cost (in terms of accuracy or reliability). In this situation, the cure (reconfiguration) would be worse than the disease (failure isolation of the primary INS).
- b. If the failed sensor is the only aiding sensor active in the filter, Level 3 reconfiguration will also disable all measurements. Is this acceptable? In this case, if all measurements are disabled, the filter will propagate INS error states which have been calibrated with questionable measurements but should be behaving in a predictable manner. Rather than risking further corruption of the state vector, it is preferable in this situation to carry out the Level 3 reconfiguration.
- c. If all measurements are disabled, the χ^2 soft failure detection test will be rendered ineffective.

Decisions on measurement reconfiguration depend on the independence (or interdependence) of individual measurements. After a measurement failure has been detected (at any level), all unfailed measurements should be tested by asking "Is it likely that the phenomenon which caused the failure could adversely affect this unfailed measurement?". A negative response would lead to no change in measurement processing. An affirmative response should lead to reconfiguration, ie. disabling of the measurement in question. This is really the classic compromise between accuracy and reliability: processing a questionable measurement will lead to higher filter accuracy (if the measurement is reliable) or increased risk of filter corruption (if the measurement is corrupted by a bad sensor). Let's look at this problem in more detail.

First of all, measurements from independent sensors (A and B) are, by definition, independent so no failure condition in sensor A will have any effect on sensor B. A failed measurement from sensor A should never lead to the reconfiguration of any measurement from (independent) sensor B. Different measurements from a particular sensor are usually interdependent but are not necessarily so. All available measurements from the sensor must be carefully analyzed to identify any dependencies. This is done for each DIINS aiding sensor below.

But, what if two sensors are not independent? For example, what about an integrated navigation system with two GPS receivers: A applies differential corrections, B does not. In principle, the same test would apply, but the result may not be very clear. This is an unusual

situation which has not been encountered to date but is possible for the future. For now, all sensors will be considered to be independent of one another.

GPS Measurement Groups

There are at least 6 groups of measurements from GPS:

1. Pseudoranges,
2. Carrier phase,
3. Time,
4. Positions,
5. Velocities,
6. Attitude.

The first 2 groups are "raw" receiver measured data; the last 3 are receiver computed quantities.

A pseudorange is a kind of range measurement from the receiver's antenna to a satellite. At any particular epoch, the receiver clock has a fixed offset from GPS time, creating a fixed range bias — and thus the "pseudo" in pseudorange. Pseudoranges from different satellites share a common receiver. Thus, a receiver failure, such as a bad clock, would affect all pseudoranges. Pseudoranges also share Control Segment errors, but these are not likely to be significant. Selective Availability is applied to individual satellites and does not result in pseudorange dependencies. In summary, pseudoranges share only common receiver errors. If the receiver suffers no failures, pseudoranges can be considered independent of one another. DIINS has a limited capability to process pseudorange measurements.

Carrier phase is a generic term which is used to identify data generated from phase measurements on the GPS signal's carrier. After the navigation message and pseudorange data have been removed from the transmitted signal, the remaining carrier can be sampled to provide data which is used to provide change of range measurements. These can be used to derive range differences or range rates from the receiver to each available satellite. Again, receiver errors may affect all phase measurements. To a large degree, phase data are independent of pseudorange data. However, receiver errors may again affect carrier phase and pseudorange data simultaneously, producing interdependence.

In summary, receiver errors may affect all raw GPS receiver measurements (pseudorange and carrier phase). If receiver errors are to be considered, all raw measurements should be considered interdependent, and all computed parameters should also be considered interdependent. If receiver errors are ignored, all raw data can be considered to be independent.

In principle, four pseudoranges in a triangulation type of process are used to compute three position coordinates plus the time bias. This process introduces significant interdependencies between position coordinates and receiver time even when the pseudoranges are independent. And, of course, position is dependent on all the pseudoranges used in the computations. In some receivers, additional data from the carrier phase may be used to smooth (ie. reduce the noise levels of) the pseudoranges. This process introduces interdependencies between positions and carrier phase.

Velocities and attitudes are both computed using carrier phase data. Again, the algorithms used to compute velocities and attitude introduce internal dependencies (eg. between north and

east velocity, and between roll and pitch). Since velocity and attitude computations both rely on phase data, they are also interdependent. Methods of combining pseudoranges and carrier phase will introduce more dependencies (eg. making position dependent on carrier phase and thus interdependent with velocity and attitude).

The following tables illustrate the two extreme possibilities. Minimum dependency is derived by assuming that receiver errors are negligible in size or probability and that there is no combining of pseudorange and phase data.

Table 4.1 Minimum GPS Dependencies

	Ps-Range	Phase	Position	Velocity	Attitude
Ps-Range	X				
Phase		X			
Position	X		X		
Velocity		X		X	X
Attitude		X		X	X

Maximum dependency is derived by assuming that receiver errors are not negligible and that pseudorange and phase data are combined using smoothing techniques or a Kalman filter, etc.

Table 4.2 Maximum GPS Dependencies

	Ps-Range	Phase	Position	Velocity	Attitude
Ps-Range	X	X			
Phase	X	X			
Position	X	X	X	X	X
Velocity	X	X	X	X	X
Attitude	X	X	X	X	X

So, let's decide exactly what reconfiguration will take place in response to different levels of GPS failure. The following applies specifically to the Trimble TASMAN receiver currently used with DIINS. DIINS cannot use phase measurements at present, so phase will not be considered. Further, assume that position, velocity and attitude measurements may be used simultaneously; but pseudoranges will never be used with position, velocity or attitude. This receiver most closely resembles the minimum dependency model. All reconfiguration is summarized in the following table. Each row represents a Group of measurements, at least one of which has failed. Each column represents the measurements which are disabled in response to that failure. The

numbers indicate the level of failure and reconfiguration: 1 is residual failure, 2 is measurement Type failure, 3 is sensor failure.

Table 4.3 Measurement Reconfiguration for the TASMAN GPS Receiver

	Ps-Range	Position	Velocity	Attitude
Ps-Range	3	—	—	—
Position	—	1,2,3	3	3
Velocity	—	3	1,2,3	2*,3
Attitude	—	3	2*,3	1,2,3

* — requires that GPS velocity and attitude measurements be used in the same filter.

Loran-C Measurement Groups

Loran-C has two basic data types: time differences (TDs) and positions which may computed in the receiver using the TDs. Conceptually, a TD is the difference in arrival times between signals from a Master and Slave station. There is one Master and up to four slaves per Loran-C chain. A receiver may track one or more chains at one time. One TD defines a hyperbolic line of position; two TDs can determine a position on the earth's surface; more TDs add redundancy. All TDs from one chain are dependent through common Master signal errors and receiver errors. TDs from different chains are dependent only through receiver errors. Loran-C latitude and longitude are dependent through common TD errors.

DIINS accepts only Loran-C TD measurements from a single chain. The measurements are statistically decorrelated for Kalman filter processing through the use of well known techniques. However, this does not remove common failures (if they are present). Let's take a quick look at a simple example to illustrate this fact:

1. There are two TDs measured from one chain.
2. A TD is assumed to be computed as the difference in times of arrival:

$$TD_i = M - S_i \quad (4.8)$$

where M is the arrival time of the Master signal, S_i is the slave's signal arrival time.

3. All times are scaled in terms of range.
4. All time measurements have the same variance, σ_r^2 .
5. All arrival time measurements are assumed to be uncorrelated.

$$C_r = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_r^2 & 0 \\ 0 & 0 & \sigma_r^2 \end{bmatrix} \quad (4.9)$$

6. There is a "failure" in the Master signal. The signal is received at $(M + \epsilon)$.

In matrix notation,

$$\begin{aligned} \begin{bmatrix} TD_1 \\ TD_2 \end{bmatrix} &= \begin{bmatrix} (M + \epsilon) - S_1 \\ (M + \epsilon) - S_2 \end{bmatrix} \\ &= \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} (M + \epsilon) \\ S_1 \\ S_2 \end{bmatrix} \end{aligned} \quad (4.10)$$

and

$$\begin{aligned} C_{TD} &= \begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} C_r \begin{bmatrix} 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 2\sigma_r^2 & \sigma_r^2 \\ \sigma_r^2 & 2\sigma_r^2 \end{bmatrix} \end{aligned} \quad (4.11)$$

Using a common technique, C_{TD} is diagonalized as follows:

$$\begin{aligned} C_{TD} &= UDU^T \\ C_{TD}' &= D = U^{-1}UDU^T U^{-T} \end{aligned} \quad (4.12)$$

In Kalman filter terms, the corresponding measurements must be transformed in a similar fashion. Assuming that C_{TD} is the Kalman filter measurement covariance matrix,

$$\begin{aligned} \vec{z}_{TD} &= \begin{bmatrix} TD_1 - \Delta R_1 \\ TD_2 - \Delta R_2 \end{bmatrix} \\ &= \begin{bmatrix} (M + \epsilon) - S_1 - \Delta R_1 \\ (M + \epsilon) - S_2 - \Delta R_2 \end{bmatrix} \end{aligned} \quad (4.13)$$

and

$$\begin{aligned} \vec{z}_{TD}' &= U^{-1} \begin{bmatrix} (M + \epsilon) - S_1 - \Delta R_1 \\ (M + \epsilon) - S_2 - \Delta R_2 \end{bmatrix} \\ &= \frac{1}{2\sigma_r^2} \begin{bmatrix} 1 & -1/2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} (M + \epsilon) - S_1 - \Delta R_1 \\ (M + \epsilon) - S_2 - \Delta R_2 \end{bmatrix} \\ &= \frac{1}{2\sigma_r^2} \begin{bmatrix} \frac{1}{2}(M + \epsilon) - S_1 - \Delta R_1 + S_2 - \Delta R_2 \\ (M + \epsilon) - S_2 - \Delta R_2 \end{bmatrix} \end{aligned} \quad (4.14)$$

All of which serves to illustrate that a failure in the Master will be seen in both TD measurements even after the TDs have been decorrelated. Unmodelled or poorly modelled signal propagation delays or cycle slips are the most likely causes of such an error. Of course, problems with a slave's signal will only be seen in the one TD.

An error in the receiver's clock rate will affect all TDs, even those from different chains. The TD error (for a constant clock rate error) would be proportional to the TD (ie. it would produce a scale factor error).

Again, there are different available courses of action:

1. Ignore possible Master signal errors and receiver clock errors and assume all TDs are independent;
2. Ignore possible receiver clock errors and assume that TDs from one chain are dependent on common Master signal errors while TDs from different chains are independent;
3. Assume that all TDs are interdependent though common receiver clock errors.

Note that if the receiver tracks only one chain, Master signal errors or receiver clock errors will result in the interdependence of all TDs.

How does one decide on the most appropriate choice? Let's look at the error sources more closely. DIINS uses an Internav LC360 Loran-C receiver. This receiver, and in fact most receivers, check their internal clock rate by measuring the so-called Group Repetition Interval (GRI). Each station in a chain transmits a series of pulses at a known and fixed GRI. The difference between the measured GRI and the known GRI provides a measure of the clock drift. If it is unacceptably high, the receiver status is set to bad. Thus, the DIINS FDI processing can assume that the Loran-C clock meets acceptable standards of performance.

Unexpected errors originating at the Loran-C transmitters are very unlikely. The most probable sources of error in a Loran-C (Master) signal are cycle slips and propagation delays. Loran-C operates around 100kHz: a cycle slip will result in sudden change in the TD(s) of about 3 km (1.6 nm). This is at least an order of magnitude larger than expected TD errors and is quite easy to identify. In DIINS, cycle slips are detected and, if possible, corrected using TD measurement residuals. The cycle slip is either successfully corrected or the residual is rejected. Significant Loran-C propagation delays result from ground wave transmission through the earth's atmosphere instead of a vacuum (Primary Phase Lag — PPL) and over water (Secondary Phase Lag — SPL) and land (Additional Secondary Phase Lag or Factor — ASL or ASF). In DIINS, PPL and SPL are combined in a single correction called the SPF. The SPF model relies on average atmospheric and sea water models which were used to generate a series of five constant coefficients. SPF depends only on propagation distance. For example, at 500 km distance, the SPF correction is 170 m for PPL and 220 m for SPL for a total SPF of 390 m.

ASF depends on the distance travelled over land, the type of terrain traversed, the electrical properties of the terrain, etc. For a given receiver location, an ASF varies only a small amount with time (due to snow cover moisture content, etc.). However, it changes greatly with position (as the propagation path changes). Thus, it is very difficult to model. It is typically estimated using calibration techniques: a vehicle equipped with Loran-C and a more accurate reference system travels throughout an area of interest. Errors in TDs are computed using the reference system and known transmitter positions. After removing PPL and SPL, any remaining error is attributed to ASF. Loran-C charts are corrected for ASF effects. Other users can estimate ASFs using polynomials, known grid points, etc. via the calibration data. In DIINS, polynomial models covering three Canadian chains (GRI 5930 — Canadian East Coast, GRI 9960 — NorthEast U.S., and GRI 5990 — Canadian West Coast) are available for ASF corrections. Other regions of earth will not have ASF corrections. Unfortunately, ASF is potentially the largest propagation effect: at extremes of distance and poor terrain, ASFs can reach 1000s of metres. More typically, ASFs are less than a few hundred metres. Clearly, they are a defining source of Loran-C error. As Loran-C was surpassed by more advanced technology (eg. GPS), it became difficult to justify spending additional funds to improve ASF calibration models. Also, common standards were never established. In fact, in Canada, west coast charts were drawn using ASFs from polynomial models and east coast charts were drawn using grid points and interpolation techniques. Therefore, it is unlikely that DIINS' ASF capability will ever be expanded.

So, we have dealt with receiver clock errors and with at least two of three signal propagation errors. What should be done about measurement reconfiguration? There are two conflicting

observations which make a determination more difficult:

1. All TDs in a particular chain will be affected by an error in a the Master signal — for example, unmodelled ASFs.
2. Assuming all TDs are independent adds significantly to the flexibility and robustness of the Loran-C filtering — for example, a single good TD can improve the filter's error estimates.

In DIINS, the second point will be given precedence and all TDs will be assumed to be independent from all others; the residual failure of any TD measurement will not have a direct impact on any other measurements.

INS Measurement Groups

The basic data from an INS are the outputs of its accelerometers and gyros. Conventional accelerometers are single axis devices and independent of one another. Gyros may be single axis or dual axis; independent or interdependent. Few INSs provide raw sensor data; internally generated navigation data is the norm. The DIINS software does not accept INS sensor outputs; the current DIINS INS does not provide this data. Processing to compute the position, velocity and attitude from accelerometer and gyro data introduces significant interdependencies in all INS navigation data.

All reconfiguration is summarized in the following table. Each row represents a Group of measurements, at least one of which has failed. Each column represents the measurements which are disabled in response to that failure. The numbers indicate the level of failure and reconfiguration: 1 is residual failure, 2 is measurement Type failure, 3 is sensor failure. Note that this table applies only to the INS2 measurements of the INS1 + INS2 filter.

Table 4.4 Measurement Reconfiguration for an INS

	Position	Velocity	Attitude
Position	1,2,3	2,3	2,3
Velocity	2,3	1,2,3	2,3
Attitude	2,3	2,3	1,2,3

Note the restrictions on primary INS reconfigurations listed at the start of this section.

Chapter 5

CONCLUSIONS

This report has explained the need for sensor Failure Detection, Isolation and Reconfiguration (FDIR), the types of failures which need to be monitored and techniques which are used to meet the requirements. In summary,

1. FDIR is needed
 - a. To protect DIINS' navigation filters from corruption due to the presence of unmodelled sensor errors;
 - b. To allow internal monitoring of sensor performance which can be used to notify the operator of possible sensor malfunction, thus permitting repair or replacement of a failed sensor;
 - c. To facilitate selection of the most reliable and most accurate navigation output source.
2. Hard failures are sudden and significant departures from expected sensor behaviour. There are 3 levels of hard FDI which are accompanied by 3 levels of measurement reconfiguration designed to protect the filters from corruption. There are two types of hard failure.
 - a. The first occurs when good data is no longer being supplied in a timely manner by the sensor. There are two sources of failure:
 - Sensor Built In Tests may identify a malfunction and send a bad status flag,
 - Or, if data is being sent at a much slower rate than expected, the DIINS Data Collection Task will set the status flag to bad and send dummy data records at the expected rate.
 - b. If the sensor status flags are OK, sensor data is used to form Kalman filter measurements. If these measurements do not agree with filter predicted values, a residual failure is declared (and Level 1 measurement reconfiguration enabled). Residual test failures may lead to measurement failures (and Level 2 measurement reconfiguration) and eventually sensor hard failure isolation (resulting in Level 3 measurement reconfiguration and an error message for the operator).
3. Soft failures grow slowly with time and, by definition, corrupt the filters using the failed sensor. A statistical test is used to monitor each filter (individual sensors are not tested). Soft failure isolation of a particular sensor results in another error message for the operator but has no other effect on the system.
4. Using the results of all the FDI tests, DIINS tries to find the most reliable and accurate filter for output. If no reliable filter is available, a reliable sensor will be sought (and the operator notified). In extreme situations, there may be no reliable output source. Again, the operator is notified.

In addition to describing the DIINS FDIR system, the appendices to this report provide

1. a cross-reference between the procedures described and the corresponding DIINS subroutine and file name
2. a number of DIINS operator logs from post-mission processing results using actual sea trial data.

References

- [1] D. A. G. Arden, "KALMAN FILTER DESIGN OF THE DUAL INERTIAL INTEGRATED NAVIGATION SYSTEM," Tech. Rep. Report No. DAC-02-97, Dale Arden Consulting, 1997. Prepared for the Defence Research Establishment Ottawa.
- [2] D. A. G. Arden, "ANALYSIS OF REAL TIME RESULTS FROM DIINS' 1995 SEA TRIALS," Tech. Rep. Report No. DAC-01-97, Dale Arden Consulting, 1997. Prepared for the Defence Research Establishment Ottawa.
- [3] J. C. McMillan, J. S. Bird, and D. A. Arden, "Techniques for Soft-Failure Detection in a Multisensor Integrated System," *NAVIGATION: Journal of the Institute of Navigation*, vol. 40, no. 3, Fall 1993.
- [4] K. R. Britting, *Inertial Navigation Systems Analysis*. New York: John Wiley & Sons, Inc., 1971.
- [5] B. Brumback and M. Srinath, "A Fault-Tolerant Multisensor Navigation System Design," *IEEE Transactions on Aerospace and Electronic Systems*, pp. 738-756, Nov. 1987.

Appendix A DIINS FDIR SOFTWARE

CROSS-REFERENCES

In DIINS, hard and soft failure detection and Levels 1 and 2 measurement reconfiguration are performed at the Kalman filter level:

1. there is one filter task initiated for each filter;
2. the Kalman filter task and all DIINS specific subroutines are found in the directory (eg. */home/Moe/dale/DIINS/UNIFILTER*;
3. all low level Kalman filter library subroutines are found in the directory (eg. */home/Moe/dale/DIINS/LIBRARY*;
4. all system definition subroutines are found in the directory (eg. */home/Moe/dale/DIINS/SYSTEM*;
5. the code for the Kalman filter task is stored in file *Prefilter.ada*;
6. the name of the Kalman filter task is *Prefilter_TaskType*;

The FDIR processor collects data from all the filters to isolate hard and soft failures and to select the best source for output:

1. there is only one FDIR task regardless of the number of filters running;
2. the FDIR task and all related subroutines are found in the directory (eg. */home/Moe/dale/DIINS/FDI*;
3. system definition subroutines are again found in the directory (eg. */home/Moe/dale/DIINS/SYSTEM*;
4. the code for the FDIR task is stored in file *FDI.ada*;
5. the name of the FDIR task is *FDI_TaskType*;

The FDIR processor returns data to each filter indicating when and how Level 3 measurement reconfiguration should be performed.

In general, subroutine names are the same as the prefix of the file in which the subroutine is stored. For example, subroutine *Kalman* is stored in the file *Kalman.ada*. In the following descriptions, subroutines will be identified using their names. If the file prefix is different (eg. *Prefilter_TaskType* is stored in file *Prefilter.ada*), the subroutine name will be given followed by the file name in brackets, eg. *Prefilter_TaskType (Prefilter.ada)*.

A.1 Hard Failure Detection and Level 1 and 2 Measurement Reconfiguration

All hard failure detection is based on the Kalman filter measurements. At the prescribed time intervals (the basic filter rate which is some integer multiple of the INS data rate) *Prefilter_TaskType (Prefilter.ada)* calls *Kalman*. Whenever measurements are to be formed (at some integer multiple of the basic filter rate), *Kalman* calls subroutine *FilterTheMeasurements*.

In *FilterTheMeasurements*, each Measurement Group is processed in a separate subroutine. Each Group is processed in a similar manner:

1. The latest data records from the primary INS and the appropriate aiding sensor are checked. This check has two parts —
 - a. The record should not have been used previously to form measurements.
 - b. And, the status word sent from the sensor's Data Collection Task (DCT) must be OK. Again, this (BIT/DCT) status word has two components. OK status will be sent only if —
 - The Built In Tests (BITs) from the sensor indicate the sensor is operating properly.
 - And, there must be no gaps in the sensor data. If the DCT detects an unacceptably large gap when no data has been received, it will set the status word to BAD, fill in its output record with the last data record collected and send this bad record on to the filters requesting it.
2. If the status is good, the difference in the time tags between the primary INS and the aiding sensor is computed. If the size of the time difference is
 - a. less than the maximum allowed,
 - increment the MeasFormed counters (in DIINS they are called *FailureIsoLimit*) for the primary INS and the aiding sensor,
 - and call the subroutine(s) to process the selected measurements.
 - b. more than the maximum allowed, the operator is notified.

Each measurement subroutine has a name of the type *INS_Aid_MeasGroup_Meas*. For example, the subroutine which processes GPS position measurements is called *INS_GPS_Pos_Meas*. The measurement subroutines compute the residuals and send them to the Kalman filter update routine, *StateVectorUpdate* one at a time. *StateVectorUpdate* calls the library subroutine *Update* where the filter is updated and the residual variance is computed. Back in *StateVectorUpdate*, the residual is tested with its variance. The results of the test are used to update the measurement history record and increment the appropriate residual pass or fail counters.

Level 1 measurement reconfiguration is performed in the measurement subroutines. It is implemented as follows:

1. If measurements within the Group are dependent:
 - a. All state information is saved as soon as the subroutine is called.
 - b. All measurements are formed, residuals calculated etc. one at a time.
 - c. After all measurements in the Group have been processed, and if any residual in the Group has failed, (or if the status of any measurement in the Group is not OK — see the *MeasFailureDetection* subroutine below), all state information is reset to the values saved at the beginning of the procedure.

Control returns to *Prefilter_TaskType(Prefilter.ad)* where the measurement failure detection subroutine, *MeasFailureDetection*, is called. *MeasFailureDetection* simply compares the residual pass and fail counters for each measurement used in the filter with predefined limits. If the limits are exceeded, a message is sent to the operator. When the fail counter for any measurement exceeds its limit,

1. a status flag for that particular measurement is set to the failed mode,
2. a flag is set to activate Level 2 measurement reconfiguration.
3. and the MeasFailed counter (in DIINS they are called *SensorFailCnt*) for the primary INS and the appropriate aiding sensor are incremented.

The status flag for a failed measurement is set to OK only after the measurement pass counter exceeds its limit.

Level 2 measurement reconfiguration is initiated in *MeasFailureDetection* by calling *ReconfigureMeas* with the *LEVEL2* argument. In response to the *LEVEL2* argument, *ReconfigureMeas* calls *GPS_Level2_MeasReconfig* and *INS_Level2_MeasReconfig*. Level 2 measurement reconfiguration is carried according to the specifications outlined in Section 4.2. Each subroutine checks to see if dependent measurements from different Groups (from GPS and INS respectively) are being used. Then, if any measurement has failed, all dependent measurements which belong to different Measurement Groups are subjected to Level 2 reconfiguration: the status of each reconfigured measurement is set to *RECONFIG_2*. (Recall that dependent measurements within a particular Group are subjected to Level 1 reconfiguration.) The same subroutines remove the all Level 2 reconfigurations after all measurements in a Group have recovered.

GPS measurement failures result in the following Level 2 measurement reconfiguration:

A failure in any GPS	Will produce Level 2 reconfiguration in all
Velocity measurement	Attitude measurements
Attitude measurement	Velocity measurements

INS measurement failures result in the following Level 2 measurement reconfiguration:

A failure in any INS	Will produce Level 2 reconfiguration in all
Position measurement	Velocity and attitude measurements
Velocity measurement	Position and attitude measurements
Attitude measurement	Position and velocity measurements

A.2 Soft Failure Detection

Soft failure detection is performed using the χ^2 test described in Section 3.1. The χ^2 test and the shadow filter used in the χ^2 test are initialized in subroutine *ChiSquaredTestInit* which called once from *Prefilter_TaskType(Prefilter.ad)*. Subroutine *ChiSquaredFilterTest* is called at the state propagation time interval. This subroutine propagates the shadow filter using the propagation model computed for the main Kalman filter. At an interval which is an integer

multiple of the Kalman filter update rate, the χ^2 test statistic is computed and tested using a χ^2 distribution look-up table. The χ^2 status flag is set to *FAILED* or *OK* depending on the results of the test. Note that if the computations used to compute the statistic have been judged to be numerically unstable as the result of a poorly conditioned matrix, the χ^2 status flag is set to *BAD_COND*.

When the χ^2 test fails (only when *ChiSquaredStatus* = *FAILED*), soft failure detection is assumed.

A.3 Hard Failure Isolation

All hard failure detection counters (*FailureIsoLimit* and *SensorFailCnt*) and the primary INS status from all filters are sent to the FDI processor (*FDI_TaskType*) to facilitate Hard Failure Isolation (HFI). Recall that there is only one of each counter for each sensor from each filter (see the discussion in Section 2.1).

The first time *FDI_TaskType* is called, subroutine *ReconfigureFromFDI* is used to initialize the best filter for output (by calling *ID_BestFilterForDisplay*) and to select the best filters for the hard failure isolation testing of each sensor (in subroutine *FindFiltersForHFI*). The procedures used to find the best output filter and best filters for HFI are detailed in Chapter 2.

Every time *FDI_TaskType* is called, the HFI testing of each sensor is performed by calling subroutine *HardSensorFailureIsolation*. This subroutine uses the HFI filters selected in *FindFiltersForHFI* and the logic of Section 2.2 to isolate hard sensor failures. For every active sensor which is NOT suffering from a soft failure, three hard failure flags are set:

1. If measurements have been successfully formed using the sensor in any HFI filter, the *MEAS_PROCESSED* flag is set to *TRUE*.
2. If all the measurements formed in all HFI filters have failed, the *HARD_FAILURE* flag is set to *TRUE*.
3. If any problem was detected, the *MEAS_FAILURE* flag is set to *TRUE*: using Boolean arithmetic, *MEAS_FAILURE* = (NOT *MEAS_PROCESSED*) OR *HARD_FAILURE*.

Immediately after calling *HardSensorFailureIsolation*, *FDI_TaskType* calls *CheckINS_StatusFailure*. This subroutine checks the BIT/DCT status flags sent from the primary INS of every filter. If any primary INS status flag is bad, that INS will be flagged. There are only two INSS, but all filters are checked in the unlikely event that different filters are using different INS records.

A.4 Soft Failure Isolation

The χ^2 test result (*ChiSquaredStatus* = *FAILED*) is sent from every filter to *FDI_TaskType* to facilitate Soft Failure Isolation (SFI). The first time through, when subroutine *ReconfigureFromFDI* is called, subroutine *FindFiltersForSFI* selects the best filters for the SFI testing of each sensor. The filters are selected according to the concepts presented in Chapter 3.

Again, every time *FDI_TaskType* is called, the SFI testing of each sensor is performed by calling subroutine *SoftSensorFailureIsolation*. This subroutine uses the SFI filters selected in

FindFiltersForSFI and the logic of Section 3.2 to isolate soft sensor failures. For every active sensor which is NOT suffering from any hard failure, the soft failure flag is set as follows:

If all the SFI filters have failed their χ^2 test, the *SOFT_FAILURE* flag is set to *TRUE*.

A.5 FDIR Reconfiguration

Whereas hard and soft sensor status is checked every time *FDI_TaskType* is called, neither the filters used for HFI and SFI nor the source for system output change until the FDI system changes. As noted above, the failure isolation filters and output source are set by calling *ReconfigureFromFDI* from *FDI_TaskType*. This done the first time *FDI_TaskType* is called, then only if the *ReconfigurationRequired* flag is set to *TRUE*. The *ReconfigurationRequired* flag is set to *TRUE* when any of the following conditions is identified.

1. A change in the χ^2 failure flag for any filter is detected in subroutine *SoftSensorFailureIsolation*.
2. A change in the BIT/DCT status of either INS is detected in subroutine *Check-INS_StatusFailure*.
3. A change in the MeasFormed (*FailureIsoLimit*) or MeasFailed (*SensorFailCnt*) counter is detected in subroutine *HardSensorFailureIsolation* under the following conditions:
 - a. More measurements have failed or fewer measurements were formed for any sensor in the filter currently used for output (this is an indication of degraded performance — maybe there is another filter which is better).
 - b. Fewer measurements have failed or more measurements were formed for any sensor in any other filter (this is an indication of improved performance — maybe this filter should now be used for output).
4. A change in the *MEAS_PROCESSED* flag for any sensor is detected in subroutine *HardSensorFailureIsolation*.
5. A change in the *HARD_FAILURE* flag for any sensor is detected in subroutine *HardSensorFailureIsolation*.
6. A change in the *SOFT_FAILURE* flag for any sensor is detected in subroutine *SoftSensorFailureIsolation*.

A.6 Level 3 Measurement Reconfiguration

At the completion of FDIR processing, the *ReconfigurationRequired* flag is sent back to every filter (where it now the *SensorStatusHasChanged* flag. Whenever this flag is true, Level 3 measurement reconfiguration is initiated in every filter: subroutine *ReconfigureMeas* is called with the *LEVEL3* argument. In *ReconfigureMeas*, Level 3 measurement reconfiguration proceeds as follows:

1. All measurements which are formed using an aiding sensor which has suffered a hard failure (ie. the *HARD_FAILURE* flag is *TRUE*) will be disabled.
2. All measurements which are formed using an aiding sensor which has recovered from a hard failure (ie. the *HARD_FAILURE* flag is *FALSE*) will be reactivated if they have been

disabled by a Level 3 reconfiguration. If the status of a particular measurement is bad for any other reason (eg. residual failures, Level 1 or 2 reconfiguration), it will not be reactivated.

Appendix B TEST RESULTS

Data collected from 2 Sperry Marine MK29 Mod 3 INSs, a Trimble TASMAN P(Y) code GPS receiver and an Internav LC360 Loran-C receiver during a sea trial on board the research ship Endeavor in May 1995 (ST95) was used to test the FDIR procedures described in this document. Four different data sets were collected:

1. Julian days 129–132 (May 9 – 12),
2. Julian day 133 (May 13),
3. Julian days 134–135 (May 14 – 15),
4. and Julian day 136 (May 16).

Each data set consists of a full DIINS run. During the trials, DIINS had to be stopped and restarted for various reasons. When DIINS was started for Day 136, the INSs were not given position fixes to reduce their position errors. As a result, there are initial position errors of about 3200 metres in latitude and 1200 m in longitude. By setting DIINS' initial INS position standard deviations to values much smaller than the true errors, various χ^2 test, measurement and sensor failures can be induced. These induced failures act to exercise several FDIR processor modes over the first 1–2 hours of the run. All failures eventually recover, thus testing the recovery capabilities of the FDI processor.

A number of different configurations were run the DIINS post-mission software to test the FDI processor as completely as possible:

1. A 4 sensor, 7 filter configuration with the following filters:
 - a. INS1 + INS2,
 - b. INS1 + GPS + Loran-C,
 - c. INS1 + GPS,
 - d. INS1 + Loran-C,
 - e. INS2 + GPS + Loran-C,
 - f. INS2 + GPS,
 - g. INS2 + Loran-C.
2. A 3 sensor, 3 filter configuration with the following filters:
 - a. INS1 + INS2,
 - b. INS1 + GPS,
 - c. INS2 + GPS,
3. A 3 sensor, 3 filter configuration with the following filters:
 - a. INS1 + INS2,
 - b. INS1 + Loran-C,
 - c. INS2 + Loran-C.
4. A 1 sensor, 1 filter configuration:
 - a. INS1 + INS2,

5. A 1 sensor, 1 filter configuration:
 - a. INS1 + GPS,
6. A 1 sensor, 1 filter configuration:
 - a. INS1 + Loran-C,
7. A 1 sensor, 1 filter configuration:
 - a. INS2 + GPS,
8. A 1 sensor, 1 filter configuration:
 - a. INS2 + Loran-C,

All FDI messages sent to the operator's screen were saved in files and are attached to this report in Sections B.1 through B.8.

The only way to test other error scenarios would involve running DIINS in debug mode and manually forcing various failure scenarios. The time consuming nature and very large number of possible scenarios precluded any attempt at this kind of testing.

B.1 Output Log for the INS1+INS2+GPS+Loran-C, 7 Filter Configuration

IRIG-B: Initializing

CollectData: BufferSize = 4096

SimConfig: Initiating...

Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.7

...Reading Sensor/Task Rates.

...Done Sensor/Task Rates.

Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-1+2-136-FDI-12GL.Report

Filter 2: Logging output to \$DIINS/RESULTS/ST95/Setup-1+PY+LC-136-FDI-12GL.Report

Filter 3: Logging output to \$DIINS/RESULTS/ST95/Setup-2+PY+LC-136-FDI-12GL.Report

Filter 4: Logging output to \$DIINS/RESULTS/ST95/Setup-1+PY-136-FDI-12GL.Report

Filter 5: Logging output to \$DIINS/RESULTS/ST95/Setup-2+PY-136-FDI-12GL.Report

Filter 6: Logging output to \$DIINS/RESULTS/ST95/Setup-1+LC-136-FDI-12GL.Report

Filter 7: Logging output to \$DIINS/RESULTS/ST95/Setup-2+LC-136-FDI-12GL.Report

Done reading config file.

DataLogger: started

Loran1_BB: Being read by 4 tasks.

GPS1_BB: Being read by 4 tasks.
GPS2_BB: Being read by 0 tasks.
GPS2_BB: Terminated Normally
LOG1_BB: Being read by 0 tasks.
LOG1_BB: Terminated Normally
Omega1_BB: Being read by 0 tasks.
Omega1_BB: Terminated Normally
INS1_BB: Being read by 4 tasks.
INS2_BB: Being read by 4 tasks.
FDI initiated
AO initiated
ASF initiated
PPC initiated
TaskMaster v.0.4
TaskMaster: Generate DCTs
TaskMaster: Generate Filter Tasks
TaskMaster: Starting Prefilter 1
Sperry_MK29_1_DCT initiated
Sperry_MK29_2_DCT initiated
Trimble_TANS_PY_DCT initiated
Internav_LC360_DCT initiated
Filter 1: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 2
Filter 2: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 3
Filter 3: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 4
Filter 4: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 5
Filter 5: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 6
Filter 6: initiated
INS-computed reset values are used to correct state vector.
TaskMaster: Starting Prefilter 7
Filter 7: initiated
INS-computed reset values are used to correct state vector.
Filter 1: Received permission to start.
Filter 2: Received permission to start.

Filter 3: Received permission to start.
Filter 4: Received permission to start.
Filter 5: Received permission to start.
Filter 6: Received permission to start.
Filter 7: Received permission to start.
FDI: Received permission to start
Sperry_MK29_1_DCT: Received permit to start
Sperry_MK29_2_DCT: Received permit to start
Trimble_TANS_PY_DCT: Received permit to start
Internav_LC360_DCT: Received permission to start
Internav_LC360_DCT: Loran-C is tracking chain GRI_5990
BBuffer 1: ReadPointer = Insert = 3 & FullSlots = 1
Filter 1: INS1 BB short
Filter 4 has been set to velocity damped mode at 74823769.092
Filter 5 has been set to velocity damped mode at 74823769.934
BBuffer 5: ReadPointer = Insert = 4 & FullSlots = 2
Filter 5: GPS1 BB short
Filter 4: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 1 has been set to velocity damped mode at 74823770.372
Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
AO: Received permission to start
AO: Terminated Normally
ASF: Received permission to start
PPC: Received permission to start
PPC: Terminated Normally
PPC: Terminated Normally
Filter 5: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 6 has been set to velocity damped mode at 74823769.092
Filter 6: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 7 has been set to velocity damped mode at 74823769.934
Filter 7: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 2 has been set to velocity damped mode at 74823769.092
Filter 2: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 3 has been set to velocity damped mode at 74823769.934
Filter 3: ChiSquared initialized to Status = BAD_COND Failure = FALSE
FDI: FilterUsedForDisplay set to 2 at time 74823771.652
SFI: Filters for
 INS1_SNSR 1 4
 INS2_SNSR 1 5
 GPS1_SNSR 4 5
 LORAN1_SNSR 6 7
HFI: Filters for
 INS1_SNSR 1 4

INS2_SNSR 1 5
GPS1_SNSR 2 3
LORAN1_SNSR 2 3
FDI: FilterUsedForDisplay stays at 2 at time 74823771.652
SFI: Filters for
INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 4 5
LORAN1_SNSR 6 7
HFI: Filters for
INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 2 3
LORAN1_SNSR 2 3
Filter 1: ChiSquared changed to Status = FAILED at time 74823774.212
Filter 3: ChiSquared changed to Status = FAILED at time 74823775.053
Filter 5: ChiSquared changed to Status = FAILED at time 74823775.053
Filter 6: ChiSquared changed to Status = OK at time 74823776.772
Filter 2: ChiSquared changed to Status = OK at time 74823776.772
Filter 7: ChiSquared changed to Status = OK at time 74823831.373
Filter 7: ChiSquared changed to Status = FAILED at time 74823862.094
Filter 6: ChiSquared changed to Status = FAILED at time 74823861.252
Filter 2: ChiSquared changed to Status = FAILED at time 74823891.971
Filter 4: measurement Y_GPS_POS_MEAS FAILURE DETECTION at time
74824037.891
Filter 6: measurement LORAN_TDZ_MEAS FAILURE DETECTION at time
74824040.452
Filter 2: measurement X_GPS_POS_MEAS FAILURE DETECTION at time
74824040.452
Filter 2: measurement LORAN_TDZ_MEAS FAILURE DETECTION at time
74824040.452
FDI: FilterUsedForDisplay set to 3 at time 74824041.293
SFI: Filters for
INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 4 5
LORAN1_SNSR 6 7
HFI: Filters for
INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 2 3
LORAN1_SNSR 2 3
Filter 1: ChiSquared changed to Failure = TRUE at time 74824076.292

Filter 5: ChiSquared changed to Failure = TRUE at time 74824077.133

Filter 3: ChiSquared changed to Failure = TRUE at time 74824077.133

FDI: INS2_SNSR SOFT failure ISOLATION at time 74824076.292

FDI: FilterUsedForDisplay set to 2 at time 74824076.292

SFI: Filters for

INS1_SNSR 1 4

INS2_SNSR 1 5

GPS1_SNSR 4 5

LORAN1_SNSR 6 7

HFI: Filters for

INS1_SNSR 4 6

INS2_SNSR 0 0

GPS1_SNSR 2 4

LORAN1_SNSR 2 6

FDI: INS1_SNSR HARD failure ISOLATION at time 74824078.852

FDI: GPS1_SNSR HARD failure ISOLATION at time 74824078.852

FDI: LORAN1_SNSR HARD failure ISOLATION at time 74824078.852

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to INS2_SNSR
at time 74824078.852

SFI: Filters for

INS1_SNSR 0 0

INS2_SNSR 0 0

GPS1_SNSR 0 0

LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 6

INS2_SNSR 0 0

GPS1_SNSR 2 4

LORAN1_SNSR 2 6

Filter 6: ChiSquared changed to Failure = TRUE at time 74824163.332

Filter 7: ChiSquared changed to Failure = TRUE at time 74824164.172

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to UNKNOWN
at time 74824163.332

SFI: Filters for

INS1_SNSR 0 0

INS2_SNSR 0 0

GPS1_SNSR 0 0

LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0

INS2_SNSR 0 0

GPS1_SNSR 2 4

LORAN1_SNSR 2 0

Filter 2: ChiSquared changed to Failure = TRUE at time 74824194.052
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74824194.052

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 0 4
LORAN1_SNSR 0 0

Filter 2: measurement X_GPS_POS_MEAS was REACTIVATED at time
74824398.852

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74824398.852

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 0 4
LORAN1_SNSR 0 0

DataLogger: Terminated Normally

Filter 6: measurement LORAN_TDZ_MEAS was REACTIVATED at time
74825202.692

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74825202.692

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 0 4
LORAN1_SNSR 0 0

Filter 2: measurement LORAN_TDZ_MEAS was REACTIVATED at time
74825230.852

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74825230.852

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 0 4
LORAN1_SNSR 0 0

Filter 4: measurement Y_GPS_POS_MEAS was REACTIVATED at time
74826265.091

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to GPS1_SNSR
at time 74826265.091

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 0 4
LORAN1_SNSR 0 0

Filter 2: ChiSquared changed to Status = OK at time 74826336.772

Filter 6: ChiSquared changed to Status = OK at time 74826434.051

Filter 2: ChiSquared changed to Failure = FALSE at time 74826638.851

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at
GPS1_SNSR at time 74826638.851

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 2 4
LORAN1_SNSR 2 0

FDI: GPS1_SNSR HARD failure RECOVERY at time 74826641.412
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at
GPS1_SNSR at time 74826641.412

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 5 0
GPS1_SNSR 0 5
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 0
INS2_SNSR 0 0
GPS1_SNSR 2 4
LORAN1_SNSR 2 0

Filter 4: ChiSquared changed to Status = OK at time 74826646.531

Filter 6: ChiSquared changed to Failure = FALSE at time 74826736.131

FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at
GPS1_SNSR at time 74826736.131

SFI: Filters for

INS1_SNSR 0 0
INS2_SNSR 5 0
GPS1_SNSR 0 5
LORAN1_SNSR 0 0

HFI: Filters for

INS1_SNSR 4 6
INS2_SNSR 0 0
GPS1_SNSR 2 4
LORAN1_SNSR 2 6

FDI: INS1_SNSR HARD failure RECOVERY at time 74826738.692

FDI: LORAN1_SNSR HARD failure RECOVERY at time 74826738.692

FDI: FilterUsedForDisplay set to 2 at time 74826738.692

SFI: Filters for

INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 4 5
LORAN1_SNSR 6 7

HFI: Filters for

INS1_SNSR 4 6
INS2_SNSR 0 0
GPS1_SNSR 2 4
LORAN1_SNSR 2 6

Filter 7: ChiSquared changed to Status = OK at time 74826852.170

Filter 1: ChiSquared changed to Status = OK at time 74826897.411

Filter 5: ChiSquared changed to Status = OK at time 74827118.410

Filter 5: ChiSquared changed to Status = FAILED at time 74827123.530
Filter 5: ChiSquared changed to Status = OK at time 74827126.089
Filter 3: ChiSquared changed to Status = OK at time 74827128.649
Filter 7: ChiSquared changed to Failure = FALSE at time 74827154.250
FDI: FilterUsedForDisplay stays at 2 at time 74827153.411
SFI: Filters for
 INS1_SNSR 1 4
 INS2_SNSR 1 5
 GPS1_SNSR 4 5
 LORAN1_SNSR 6 7
HFI: Filters for
 INS1_SNSR 4 6
 INS2_SNSR 0 0
 GPS1_SNSR 2 4
 LORAN1_SNSR 2 6
Filter 1: ChiSquared changed to Failure = FALSE at time 74827199.491
FDI: FilterUsedForDisplay stays at 2 at time 74827199.491
SFI: Filters for
 INS1_SNSR 1 4
 INS2_SNSR 1 5
 GPS1_SNSR 4 5
 LORAN1_SNSR 6 7
HFI: Filters for
 INS1_SNSR 4 6
 INS2_SNSR 0 0
 GPS1_SNSR 2 4
 LORAN1_SNSR 2 6
Filter 5: ChiSquared changed to Failure = FALSE at time 74827428.170
FDI: INS2_SNSR SOFT failure RECOVERY at time 74827427.330
FDI: FilterUsedForDisplay stays at 2 at time 74827427.330
SFI: Filters for
 INS1_SNSR 1 4
 INS2_SNSR 1 5
 GPS1_SNSR 4 5
 LORAN1_SNSR 6 7
HFI: Filters for
 INS1_SNSR 1 4
 INS2_SNSR 1 5
 GPS1_SNSR 2 5
 LORAN1_SNSR 2 7
Filter 3: ChiSquared changed to Failure = FALSE at time 74827430.729
FDI: FilterUsedForDisplay stays at 2 at time 74827429.891
SFI: Filters for

INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 4 5
LORAN1_SNSR 6 7

HFI: Filters for

INS1_SNSR 1 4
INS2_SNSR 1 5
GPS1_SNSR 2 3
LORAN1_SNSR 2 3

Filter 3 has been set to velocity UNdamped mode at 74830239.046
Filter 5 has been set to velocity UNdamped mode at 74830239.046
Filter 1 has been set to velocity UNdamped mode at 74830239.490
Filter 7 has been set to velocity UNdamped mode at 74830239.046
Filter 4 has been set to velocity UNdamped mode at 74830464.771
Filter 6 has been set to velocity UNdamped mode at 74830464.771
Filter 2 has been set to velocity UNdamped mode at 74830464.771
Filter 4 has been set to velocity damped mode at 74830702.850
Filter 6 has been set to velocity damped mode at 74830702.850
Filter 2 has been set to velocity damped mode at 74830702.850
INS1_BB: Shutdown by DCT

B.2 Output Log for the INS1+INS2+GPS, 3 Filter Configuration

IRIG-B: Initializing

CollectData: BufferSize = 4096

SimConfig: Initiating...

Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.3

...Reading Sensor/Task Rates.

...Done Sensor/Task Rates.

Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-1+2-136-FDI-12G.Report

Filter 2: Logging output to \$DIINS/RESULTS/ST95/Setup-1+PY-136-FDI-12G.Report

Filter 3: Logging output to \$DIINS/RESULTS/ST95/Setup-2+PY-136-FDI-12G.Report

Done reading config file.

DataLogger: started

Loran1_BB: Being read by 0 tasks.

Loran1_BB: Terminated Normally

GPS1_BB: Being read by 2 tasks.

GPS2_BB: Being read by 0 tasks.

GPS2_BB: Terminated Normally

LOG1_BB: Being read by 0 tasks.

LOG1_BB: Terminated Normally
 Omega1_BB: Being read by 0 tasks.
 Omega1_BB: Terminated Normally
 INS1_BB: Being read by 2 tasks.
 INS2_BB: Being read by 2 tasks.
 FDI initiated
 AO initiated
 ASF initiated
 PPC initiated
 TaskMaster v.0.4
 TaskMaster: Generate DCTs
 TaskMaster: Generate Filter Tasks
 TaskMaster: Starting Prefilter 1
 Sperry_MK29_1_DCT initiated
 Sperry_MK29_2_DCT initiated
 Trimble_TANS_PY_DCT initiated
 Filter 1: initiated
 INS-computed reset values are used to correct state vector.
 TaskMaster: Starting Prefilter 2
 Filter 2: initiated
 INS-computed reset values are used to correct state vector.
 TaskMaster: Starting Prefilter 3
 Filter 3: initiated
 INS-computed reset values are used to correct state vector.
 Filter 1: Received permission to start.
 Filter 2: Received permission to start.
 Filter 3: Received permission to start.
 FDI: Received permission to start
 Sperry_MK29_1_DCT: Received permit to start
 Sperry_MK29_2_DCT: Received permit to start
 Trimble_TANS_PY_DCT: Received permit to start
 BBuffer 1: ReadPointer = Insert = 3 & FullSlots = 1
 Filter 1: INS1 BB short
 Filter 2 has been set to velocity damped mode at 74823769.092
 Filter 3 has been set to velocity damped mode at 74823769.934
 Filter 2: ChiSquared initialized to Status = BAD_COND Failure = FALSE
 Filter 3: ChiSquared initialized to Status = BAD_COND Failure = FALSE
 Filter 1 has been set to velocity damped mode at 74823770.372
 Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
 ### FDI: FilterUsedForDisplay set to 2 at time 74823771.652
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3

GPS1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 ### FDI: FilterUsedForDisplay stays at 2 at time 74823771.652
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 Filter 1: ChiSquared changed to Status = FAILED at time 74823774.212
 Filter 3: ChiSquared changed to Status = FAILED at time 74823775.053
 Filter 2: measurement Y_GPS_POS_MEAS FAILURE DETECTION at time
 74824037.891
 ### FDI: FilterUsedForDisplay set to 3 at time 74824038.733
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 Filter 1: ChiSquared changed to Failure = TRUE at time 74824076.292
 Filter 3: ChiSquared changed to Failure = TRUE at time 74824077.133
 ### FDI: INS2_SNSR SOFT failure ISOLATION at time 74824076.292
 ### FDI: FilterUsedForDisplay set to 2 at time 74824076.292
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 GPS1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 2 0
 INS2_SNSR 0 0
 GPS1_SNSR 2 0
 DataLogger: Terminated Normally
 Filter 2: measurement Y_GPS_POS_MEAS was REACTIVATED at time
 74826257.412
 Filter 2: ChiSquared changed to Status = FAILED at time 74826259.971

Filter 2: ChiSquared changed to Failure = TRUE at time 74826562.052
FDI: INS1_SNSR SOFT failure ISOLATION at time 74826562.052
FDI: GPS1_SNSR SOFT failure ISOLATION at time 74826562.052
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74826562.052

SFI: Filters for
INS1_SNSR 1 0
INS2_SNSR 1 0
GPS1_SNSR 2 3

HFI: Filters for
INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0

Filter 2: ChiSquared changed to Status = OK at time 74826574.852
Filter 2: ChiSquared changed to Failure = FALSE at time 74826876.931
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to GPS1_SNSR
at time 74826876.931

SFI: Filters for
INS1_SNSR 1 0
INS2_SNSR 1 0
GPS1_SNSR 2 3

HFI: Filters for
INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0

Filter 1: ChiSquared changed to Status = OK at time 74826897.411
Filter 3: ChiSquared changed to Status = OK at time 74827074.890
Filter 3: ChiSquared changed to Status = FAILED at time 74827080.010
Filter 3: ChiSquared changed to Status = OK at time 74827092.810
Filter 1: ChiSquared changed to Failure = FALSE at time 74827199.491
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at
GPS1_SNSR at time 74827199.491

SFI: Filters for
INS1_SNSR 1 0
INS2_SNSR 1 0
GPS1_SNSR 2 3

HFI: Filters for
INS1_SNSR 0 0
INS2_SNSR 0 0
GPS1_SNSR 0 0

Filter 3: ChiSquared changed to Failure = FALSE at time 74827394.889
FDI: GPS1_SNSR SOFT failure RECOVERY at time 74827394.051

```

### FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at
GPS1_SNSR at time 74827394.051
SFI: Filters for
    INS1_SNSR 1 2
    INS2_SNSR 1 3
    GPS1_SNSR 2 3
HFI: Filters for
    INS1_SNSR 0 0
    INS2_SNSR 0 0
    GPS1_SNSR 0 0
### FDI: INS1_SNSR SOFT failure RECOVERY at time 74827396.611
### FDI: INS2_SNSR SOFT failure RECOVERY at time 74827396.611
### FDI: FilterUsedForDisplay set to 2 at time 74827396.611
SFI: Filters for
    INS1_SNSR 1 2
    INS2_SNSR 1 3
    GPS1_SNSR 2 3
HFI: Filters for
    INS1_SNSR 1 2
    INS2_SNSR 1 3
    GPS1_SNSR 2 3
INS1_BB: Shutdown by DCT

```

B.3 Output Log for the INS1+INS2+Loran-C, 3 Filter Configuration

```

IRIG-B: Initializing
CollectData: BufferSize = 4096
SimConfig: Initiating...
    Enter the name of the Configuration Setup file: [$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.3
...Reading Sensor/Task Rates.
...Done Sensor/Task Rates.
Filter 1: Logging output to $DIINS/RESULTS/ST95/Setup-1+2-136-FDI-12L.Report
Filter 2: Logging output to $DIINS/RESULTS/ST95/Setup-1+PY-136-FDI-12L.Report
Filter 3: Logging output to $DIINS/RESULTS/ST95/Setup-2+PY-136-FDI-12L.Report
Done reading config file.
DataLogger: started
Loran1_BB: Being read by 2 tasks.
GPS1_BB: Being read by 0 tasks.
GPS1_BB: Terminated Normally
GPS2_BB: Being read by 0 tasks.

```

GPS2_BB: Terminated Normally
 LOG1_BB: Being read by 0 tasks.
 LOG1_BB: Terminated Normally
 Omega1_BB: Being read by 0 tasks.
 Omega1_BB: Terminated Normally
 INS1_BB: Being read by 2 tasks.
 INS2_BB: Being read by 2 tasks.
 FDI initiated
 AO initiated
 ASF initiated
 PPC initiated
 TaskMaster v.0.4
 TaskMaster: Generate DCTs
 TaskMaster: Generate Filter Tasks
 TaskMaster: Starting Prefilter 1
 Sperry_MK29_1_DCT initiated
 Sperry_MK29_2_DCT initiated
 Internav_LC360_DCT initiated
 Filter 1: initiated
 INS-computed reset values are used to correct state vector.
 TaskMaster: Starting Prefilter 2
 Filter 2: initiated
 INS-computed reset values are used to correct state vector.
 TaskMaster: Starting Prefilter 3
 Filter 3: initiated
 INS-computed reset values are used to correct state vector.
 Filter 1: Received permission to start.
 Filter 2: Received permission to start.
 Filter 3: Received permission to start.
 FDI: Received permission to start
 Sperry_MK29_1_DCT: Received permit to start
 Sperry_MK29_2_DCT: Received permit to start
 Internav_LC360_DCT: Received permission to start
 Internav_LC360_DCT: Loran-C is tracking chain GRI_5990
 BBuffer 1: ReadPointer = Insert = 3 & FullSlots = 1
 Filter 1: INS1 BB short
 AO: Received permission to start
 AO: Terminated Normally
 ASF: Received permission to start
 PPC: Received permission to start
 PPC: Terminated Normally
 Filter 2 has been set to velocity damped mode at 74823769.092
 Filter 1 has been set to velocity damped mode at 74823770.372

Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 2: ChiSquared initialized to Status = BAD_COND Failure = FALSE
Filter 3 has been set to velocity damped mode at 74823769.934
Filter 3: ChiSquared initialized to Status = BAD_COND Failure = FALSE

FDI: FilterUsedForDisplay set to 2 at time 74823771.652

SFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

HFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

FDI: FilterUsedForDisplay stays at 2 at time 74823771.652

SFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

HFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

Filter 1: ChiSquared changed to Status = FAILED at time 74823774.212

Filter 2: ChiSquared changed to Status = OK at time 74823776.772

Filter 3: ChiSquared changed to Status = OK at time 74823831.373

Filter 2: ChiSquared changed to Status = FAILED at time 74823861.252

Filter 3: ChiSquared changed to Status = FAILED at time 74823862.094

Filter 2: measurement LORAN_TDZ_MEAS FAILURE DETECTION at time
74824040.452

FDI: FilterUsedForDisplay set to 3 at time 74824041.293

SFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

HFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3
LORAN1_SNSR 2 3

Filter 1: ChiSquared changed to Failure = TRUE at time 74824076.292

FDI: FilterUsedForDisplay stays at 3 at time 74824077.133

SFI: Filters for

INS1_SNSR 1 2
INS2_SNSR 1 3

LORAN1_SNSR 2 3
HFI: Filters for
 INS1_SNSR 2 0
 INS2_SNSR 3 0
 LORAN1_SNSR 2 3
Filter 2: ChiSquared changed to Failure = TRUE at time 74824163.332
Filter 3: ChiSquared changed to Failure = TRUE at time 74824164.172
FDI: INS1_SNSR SOFT failure ISOLATION at time 74824163.332
FDI: INS2_SNSR SOFT failure ISOLATION at time 74824163.332
FDI: LORAN1_SNSR SOFT failure ISOLATION at time 74824163.332
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74824163.332
SFI: Filters for
 INS1_SNSR 1 0
 INS2_SNSR 1 0
 LORAN1_SNSR 2 3
HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
DataLogger: Terminated Normally
Filter 2: ChiSquared changed to Status = OK at time 74826564.611
Filter 2: measurement LORAN_TDZ_MEAS was REACTIVATED at time
74826585.091
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at UNKNOWN
at time 74826585.091
SFI: Filters for
 INS1_SNSR 1 0
 INS2_SNSR 1 0
 LORAN1_SNSR 2 3
HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
Filter 3: ChiSquared changed to Status = OK at time 74826847.050
Filter 3: ChiSquared changed to Status = FAILED at time 74826849.609
Filter 3: ChiSquared changed to Status = OK at time 74826852.170
Filter 2: ChiSquared changed to Failure = FALSE at time 74826866.691
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to INS1_SNSR
at time 74826866.691
SFI: Filters for
 INS1_SNSR 1 0
 INS2_SNSR 1 0

LORAN1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
 Filter 1: ChiSquared changed to Status = OK at time 74826897.411
 Filter 3: ChiSquared changed to Failure = FALSE at time 74827154.250
 ### FDI: LORAN1_SNSR SOFT failure RECOVERY at time 74827153.411
 ### FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at INS1_SNSR
 at time 74827153.411
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 LORAN1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
 Filter 1: ChiSquared changed to Failure = FALSE at time 74827199.491
 ### FDI: INS1_SNSR SOFT failure RECOVERY at time 74827199.491
 ### FDI: INS2_SNSR SOFT failure RECOVERY at time 74827199.491
 ### FDI: FilterUsedForDisplay set to 2 at time 74827199.491
 SFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 LORAN1_SNSR 2 3
 HFI: Filters for
 INS1_SNSR 1 2
 INS2_SNSR 1 3
 LORAN1_SNSR 2 3
 INS1_BB: Shutdown by DCT

B.4 Output Log for the INS1+INS2, 1 Filter Configuration

IRIG-B: Initializing
 CollectData: BufferSize = 4096
 SimConfig: Initiating...
 Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Read-
 ing CFG/SimSetup.cfg.1
 ...Reading Sensor/Task Rates.
 ...Done Sensor/Task Rates.
 Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-136-FDI-12.Report
 Done reading config file.

DataLogger: started
 Loran1_BB: Being read by 0 tasks.
 Loran1_BB: Terminated Normally
 GPS1_BB: Being read by 0 tasks.
 GPS1_BB: Terminated Normally
 GPS2_BB: Being read by 0 tasks.
 GPS2_BB: Terminated Normally
 LOG1_BB: Being read by 0 tasks.
 LOG1_BB: Terminated Normally
 Omega1_BB: Being read by 0 tasks.
 Omega1_BB: Terminated Normally
 INS1_BB: Being read by 1 tasks.
 INS2_BB: Being read by 1 tasks.
 FDI initiated
 AO initiated
 ASF initiated
 PPC initiated
 TaskMaster v.0.4
 TaskMaster: Generate DCTs
 TaskMaster: Generate Filter Tasks
 TaskMaster: Starting Prefilter 1
 Sperry_MK29_1_DCT initiated
 Sperry_MK29_2_DCT initiated
 Filter 1: initiated
 INS-computed reset values are used to correct state vector.
 Filter 1: Received permission to start.
 FDI: Received permission to start
 Sperry_MK29_1_DCT: Received permit to start
 Sperry_MK29_2_DCT: Received permit to start
 Filter 1 has been set to velocity damped mode at 74823770.372
 Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
 ### FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to INS1_SNSR
 at time 74823771.652
 SFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 Filter 1: ChiSquared changed to Status = FAILED at time 74823774.212
 Filter 1: ChiSquared changed to Failure = TRUE at time 74824076.292
 ### FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at INS1_SNSR
 at time 74824076.292

SFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 DataLogger: Terminated Normally
 Filter 1: ChiSquared changed to Status = OK at time 74826897.411
 Filter 1: ChiSquared changed to Failure = FALSE at time 74827199.491
 ### FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay stays at INS1_SNSR
 at time 74827199.491
 SFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 HFI: Filters for
 INS1_SNSR 0 0
 INS2_SNSR 0 0
 INS1_BB: Shutdown by DCT

B.5 Output Log for the INS1+GPS, 1 Filter Configuration

IRIG-B: Initializing
 CollectData: BufferSize = 4096
 SimConfig: Initiating...
 Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.1
 ...Reading Sensor/Task Rates.
 ...Done Sensor/Task Rates.
 Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-136-FDI-1G.Report
 Done reading config file.
 DataLogger: started
 Loran1_BB: Being read by 0 tasks.
 Loran1_BB: Terminated Normally
 GPS1_BB: Being read by 1 tasks.
 GPS2_BB: Being read by 0 tasks.
 GPS2_BB: Terminated Normally
 LOG1_BB: Being read by 0 tasks.
 LOG1_BB: Terminated Normally
 Omega1_BB: Being read by 0 tasks.
 Omega1_BB: Terminated Normally
 INS1_BB: Being read by 1 tasks.
 INS2_BB: Being read by 0 tasks.
 INS2_BB: Terminated Normally

FDI initiated
AO initiated
ASF initiated
PPC initiated
TaskMaster v.0.4
TaskMaster: Generate DCTs
TaskMaster: Generate Filter Tasks
TaskMaster: Starting Prefilter 1
Sperry_MK29_1_DCT initiated
Trimble_TANS_PY_DCT initiated
Filter 1: initiated
INS-computed reset values are used to correct state vector.
Filter 1: Received permission to start.
FDI: Received permission to start
Sperry_MK29_1_DCT: Received permit to start
Trimble_TANS_PY_DCT: Received permit to start
Filter 1 has been set to velocity damped mode at 74823769.092
Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
FDI: FilterUsedForDisplay set to 1 at time 74823771.652
SFI: Filters for
 INS1_SNSR 0 1
 GPS1_SNSR 1 0
HFI: Filters for
 INS1_SNSR 0 0
 GPS1_SNSR 0 0
Filter 1: measurement Y_GPS_POS_MEAS FAILURE DETECTION at time
74824037.891
FDI: FilterUsedForDisplay stays at 1 at time 74824037.891
SFI: Filters for
 INS1_SNSR 0 1
 GPS1_SNSR 1 0
HFI: Filters for
 INS1_SNSR 0 0
 GPS1_SNSR 0 0
Filter 1: measurement Y_GPS_POS_MEAS was REACTIVATED at time
74826257.412
Filter 1: ChiSquared changed to Status = FAILED at time 74826259.971
Filter 1: ChiSquared changed to Failure = TRUE at time 74826562.052
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to GPS1_SNSR
at time 74826562.052
SFI: Filters for
 INS1_SNSR 0 1
 GPS1_SNSR 1 0

HFI: Filters for

INS1_SNSR 0 0

GPS1_SNSR 0 0

Filter 1: ChiSquared changed to Status = OK at time 74826574.852

Filter 1: ChiSquared changed to Failure = FALSE at time 74826876.931

FDI: FilterUsedForDisplay set to 1 at time 74826876.931

SFI: Filters for

INS1_SNSR 0 1

GPS1_SNSR 1 0

HFI: Filters for

INS1_SNSR 0 0

GPS1_SNSR 0 0

DataLogger: Terminated Normally

INS1_BB: Shutdown by DCT

B.6 Output Log for the INS1+Loran-C, 1 Filter Configuration

Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.1

Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-136-FDI-1L.Report

Done reading config file.

DataLogger: started

Loran1_BB: Being read by 1 tasks.

GPS1_BB: Being read by 0 tasks.

GPS1_BB: Terminated Normally

GPS2_BB: Being read by 0 tasks.

GPS2_BB: Terminated Normally

LOG1_BB: Being read by 0 tasks.

LOG1_BB: Terminated Normally

INS1_BB: Being read by 1 tasks.

INS2_BB: Being read by 0 tasks.

INS2_BB: Terminated Normally

FDI initiated

AO initiated

ASF initiated

PPC initiated

TaskMaster v.0.4

TaskMaster: Generate DCTs

TaskMaster: Generate Filter Tasks

TaskMaster: Starting Prefilter 1

Sperry_MK29_1_DCT initiated

Internav_LC360_DCT initiated

Filter 1: initiated

INS-computed reset values are used to correct state vector.
Filter 1: Received permission to start.
FDI: Received permission to start
Sperry_MK29_1_DCT: Received permit to start
Internav_LC360_DCT: Received permission to start
Internav_LC360_DCT: Loran-C is tracking chain GRI_5990
AO: Received permission to start
AO: Terminated Normally
ASF: Received permission to start
PPC: Received permission to start
PPC: Terminated Normally
Filter 1 has been set to velocity damped mode at 74823769.092
Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
FDI: FilterUsedForDisplay set to 1 at time 74823771.652
SFI: Filters for
 INS1_SNSR 0 1
 LORAN1_SNSR 1 0
HFI: Filters for
 INS1_SNSR 0 0
 LORAN1_SNSR 0 0
Filter 1: ChiSquared changed to Status = OK at time 74823776.772
Filter 1: ChiSquared changed to Status = FAILED at time 74823861.252
Filter 1: measurement LORAN_TDZ_MEAS FAILURE DETECTION at time
74824040.452
FDI: FilterUsedForDisplay stays at 1 at time 74824040.452
SFI: Filters for
 INS1_SNSR 0 1
 LORAN1_SNSR 1 0
HFI: Filters for
 INS1_SNSR 0 0
 LORAN1_SNSR 0 0
Filter 1: ChiSquared changed to Failure = TRUE at time 74824163.332
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to INS1_SNSR
at time 74824163.332
SFI: Filters for
 INS1_SNSR 0 1
 LORAN1_SNSR 1 0
HFI: Filters for
 INS1_SNSR 0 0
 LORAN1_SNSR 0 0
DataLogger: Terminated Normally
Filter 1: ChiSquared changed to Status = OK at time 74826567.171

Filter 1: measurement LORAN_TDZ_MEAS was REACTIVATED at time 74826613.251

Filter 1: ChiSquared changed to Failure = FALSE at time 74826869.251

FDI: FilterUsedForDisplay set to 1 at time 74826869.251

SFI: Filters for

INS1_SNSR 0 1

LORAN1_SNSR 1 0

HFI: Filters for

INS1_SNSR 0 0

LORAN1_SNSR 0 0

INS1_BB: Shutdown by DCT

B.7 Output Log for the INS2+GPS, 1 Filter Configuration

IRIG-B: Initializing

CollectData: BufferSize = 4096

SimConfig: Initiating...

Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.1

...Reading Sensor/Task Rates.

...Done Sensor/Task Rates.

Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-136-FDI-2G.Report

Done reading config file.

DataLogger: started

Loran1_BB: Being read by 0 tasks.

Loran1_BB: Terminated Normally

GPS1_BB: Being read by 1 tasks.

GPS2_BB: Being read by 0 tasks.

GPS2_BB: Terminated Normally

LOG1_BB: Being read by 0 tasks.

LOG1_BB: Terminated Normally

Omega1_BB: Being read by 0 tasks.

Omega1_BB: Terminated Normally

INS1_BB: Being read by 0 tasks.

INS1_BB: Terminated Normally

INS2_BB: Being read by 1 tasks.

FDI initiated

AO initiated

ASF initiated

PPC initiated

TaskMaster v.0.4

TaskMaster: Generate DCTs

TaskMaster: Generate Filter Tasks

TaskMaster: Starting Prefilter 1
Sperry_MK29_2_DCT initiated
Trimble_TANS_PY_DCT initiated
Filter 1: initiated
INS-computed reset values are used to correct state vector.
Filter 1: Received permission to start.
FDI: Received permission to start
Sperry_MK29_2_DCT: Received permit to start
Trimble_TANS_PY_DCT: Received permit to start
Filter 1 has been set to velocity damped mode at 74823769.934
Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
FDI: FilterUsedForDisplay set to 1 at time 74823772.493
SFI: Filters for
 INS2_SNSR 0 1
 GPS1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 GPS1_SNSR 0 0
Filter 1: ChiSquared changed to Status = FAILED at time 74823775.053
Filter 1: ChiSquared changed to Failure = TRUE at time 74824077.133
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to GPS1_SNSR
at time 74824077.133
SFI: Filters for
 INS2_SNSR 0 1
 GPS1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 GPS1_SNSR 0 0
DataLogger: Terminated Normally
Filter 1: ChiSquared changed to Status = OK at time 74827074.890
Filter 1: ChiSquared changed to Status = FAILED at time 74827080.010
Filter 1: ChiSquared changed to Status = OK at time 74827092.810
Filter 1: ChiSquared changed to Failure = FALSE at time 74827394.889
FDI: FilterUsedForDisplay set to 1 at time 74827394.889
SFI: Filters for
 INS2_SNSR 0 1
 GPS1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 GPS1_SNSR 0 0
INS2_BB: Shutdown by DCT

B.8 Output Log for the INS2+Loran-C, 1 Filter Configuration

IRIG-B: Initializing
CollectData: BufferSize = 4096
SimConfig: Initiating...
Enter the name of the Configuration Setup file: [\$DIINS/CFG/SimSetup.cfg] Reading CFG/SimSetup.cfg.1
...Reading Sensor/Task Rates.
...Done Sensor/Task Rates.
Filter 1: Logging output to \$DIINS/RESULTS/ST95/Setup-136-FDI-2L.Report
Done reading config file.
DataLogger: started
Loran1_BB: Being read by 1 tasks.
GPS1_BB: Being read by 0 tasks.
GPS1_BB: Terminated Normally
GPS2_BB: Being read by 0 tasks.
GPS2_BB: Terminated Normally
LOG1_BB: Being read by 0 tasks.
LOG1_BB: Terminated Normally
Omega1_BB: Being read by 0 tasks.
Omega1_BB: Terminated Normally
INS1_BB: Being read by 0 tasks.
INS1_BB: Terminated Normally
INS2_BB: Being read by 1 tasks.
FDI initiated
AO initiated
ASF initiated
PPC initiated
TaskMaster v.0.4
TaskMaster: Generate DCTs
TaskMaster: Generate Filter Tasks
TaskMaster: Starting Prefilter 1
Sperry_MK29_2_DCT initiated
Internav_LC360_DCT initiated
Filter 1: initiated
INS-computed reset values are used to correct state vector.
Filter 1: Received permission to start.
FDI: Received permission to start
Sperry_MK29_2_DCT: Received permit to start
Internav_LC360_DCT: Received permission to start
Internav_LC360_DCT: Loran-C is tracking chain GRI_5990
AO: Received permission to start
AO: Terminated Normally
ASF: Received permission to start

PPC: Received permission to start
PPC: Terminated Normally
Filter 1 has been set to velocity damped mode at 74823769.934
Filter 1: ChiSquared initialized to Status = BAD_COND Failure = FALSE
FDI: FilterUsedForDisplay set to 1 at time 74823772.493
SFI: Filters for
 INS2_SNSR 0 1
 LORAN1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
Filter 1: ChiSquared changed to Status = OK at time 74823831.373
Filter 1: ChiSquared changed to Status = FAILED at time 74823862.094
Filter 1: ChiSquared changed to Failure = TRUE at time 74824164.172
FDI: NO FILTER FOR DISPLAY - SensorUsedForDisplay set to INS2_SNSR
at time 74824164.172
SFI: Filters for
 INS2_SNSR 0 1
 LORAN1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
DataLogger: Terminated Normally
Filter 1: ChiSquared changed to Status = OK at time 74826844.490
Filter 1: ChiSquared changed to Failure = FALSE at time 74827146.570
FDI: FilterUsedForDisplay set to 1 at time 74827146.570
SFI: Filters for
 INS2_SNSR 0 1
 LORAN1_SNSR 0 1
HFI: Filters for
 INS2_SNSR 0 0
 LORAN1_SNSR 0 0
INS2_BB: Shutdown by DCT

Appendix C ADDING DIFFERENTIAL GPS AS A THIRD AIDING SENSOR

The following discussion is largely the text of a memo sent to Jeff Bird on October 18, 1996.

Adding a differential GPS receiver to the existing DIINS aiding sensor suite (P(Y)—code GPS and Loran-C) was identified as a priority item at the time the memo was written. With three aiding sensors, the standard DIINS configuration would require 15 filters running concurrently: the INS1+INS2 (FDIR only) filter plus 7 filters based on each of 2 INSs. Each INS branch would consist of the following filters:

1. A unifier using all three aids — dGPS, PY and Loran-C;
2. A second level of 3 filters, each of which excludes one aid —
 - a. dGPS+PY,
 - b. dGPS+LC,
 - c. PY+LC;
3. A third level of 3 filters, each of which excludes two aids —
 - a. dGPS only,
 - b. PY only,
 - c. LC only.

This 15 filter configuration was run using 1995 sea trial data. All filters and the FDIR appeared to perform according to expectations. This post-mission processing of two representative data sets required 105% to 113% of data time to run on a Sparc10 workstation with filter updates every 2.56 seconds. At present, the real time DIINS eXperimental Development Model (XDM) can run this configuration at only a 8.96 second rate. The XDM uses one Motorola 68040 CPU with some I/O assistance from a "smart" I/O card. There are a number of hardware and operational options which could be used to increase the filter update rate:

1. Use a faster CPU and possibly more than one CPU.
2. Other filter processing rates such as output rates, χ^2 shadow filter rates, etc.) can be increased or decreased depending on CPU capabilities and performance requirements.
3. Eliminate filters which do not add significantly to the FDIR performance.

Other options are likely possible. The next level of development on DIINS — the Advanced Development Model — will have to address this question.

Even though correlations are likely to be present between the two GPS receivers, they will be ignored in DIINS. These correlations would have to be based on the specific satellites used by each receiver, the methods used to compute positions (eg. dGPS, PPS, SPS), multipath susceptibility of each antenna, etc. Modelling them would require significant engineering and software effort.

Under the assumption that the XDM will NOT be able to adequately handle the current 15 filter configuration, what is the best way to reduce the number of filters. Here are some possibilities:

1. Collect data from both GPS receivers but only process data from the "best" one. This would be similar to the current 7 filter configuration, except that the second GPS sensor could be swapped in if the first GPS failed. When switching GPS sensors, the pertinent filter states would have to reset, GPS measurement and state error models would have to be modified, FDIR counters in the affected filters would have to be reset, etc. .
2. Alternatively, it may be possible to reduce the number of filters by empirically selecting filters for (or removing filters from) the standard 15 filter set. This approach would not require any software modifications. For example:
 - a. In the no fail situation,
 - The INS1+INS2, INS1(2)+dGPS, INS1(2)+PY and INS1(2)+LC filters are used for soft failure isolation;
 - The INS1+INS2, INS1(2)+dGPS, and the unifiers are used for hard failure isolation;
 - One of the unifiers is used for output.

All other filters could be removed. In this case, all second level filters (each with two aids) would be eliminated. Six filters would be removed, nine would remain. However, certain sensor failures will cause the unifiers to be unacceptable for output. And, since one complete level of filters has been removed, the filter selected for output after the unifiers have been rejected will be suboptimal. For example, if dGPS suffers a soft failure, the unifiers will be rejected for use as output. In the 15-filter system, the FDIR task would select one of the INS+PY+LC filters. In the 9-filter system, the FDIR task would select one of the INS+PY filters. In both systems, a subsequent PY failure could then cause FDIR to select one of the INS+LC filters for output. Note that the FAIL-FAIL-SAFE capability of the 3-aid system is maintained in this 9-filter system.

- b. Remove all filters which use both GPS sensors. This would eliminate 4 of the 15 filters, including the large unifiers (removing one complete filter level — the highest, unifier, level in this case — plus part of the second level). If the accuracy expected from both receivers is similar (eg. dGPS and PPS) and if their errors are highly correlated, there would be little to gain in terms of filter accuracy. And, the problem of modelling inter-sensor correlations would be removed. However, the degree of correlation will depend on the satellites being used by each receiver, dGPS baseline lengths, receiver technologies, etc.

Also, hard FDI of the GPS sensors may be improved by using both GPS sensors in the same filters: in general, filter states used to model the systematic errors of an aiding sensor will only be observable if another more accurate sensor is used in the same filter. If systematic errors are not modelled or not observable, they will be absorbed by other states and the measurement residuals. If the component of systematic error absorbed by the residuals has not been included in the measurement noise model (even though it is not noise), there will be an increased risk of those residuals failing. The most accurate sensor in any filter will therefore be more susceptible to hard failure false alarms. If

there are no filters using both GPS sensors, the systematic errors of the least accurate will not be observable in any filter, thus reducing the reliability of hard FDI. The most accurate sensor would not be affected very much.

Using the dGPS, PY, Loran-C system anticipated for DIINS, the following 11 filters would be run:

- INS1+INS2,
- INS1(2)+dGPS+LC,
- INS1(2)+PY+LC,
- INS1(2)+dGPS,
- INS1(2)+PY,
- INS1(2)+LC.

The FDIR task would select one of the INS1(2)+dGPS+LC filters in the no fail or PY fail environment. In a dGPS fail environment, a INS1(2)+PY+LC filter would be selected. In a Loran-C fail environment, a INS1(2)+dGPS filter would be selected. The failure of any two aids would result in the selection of an INS1(2)+Unfailed Aid filter. Again, the FAIL-FAIL-SAFE capability is maintained.

There have been suggestions that the results from the two GPS receivers could be "blended" before being sent for filtering. However, this approach has some problems:

1. It would require an additional task to collect data from both receivers, perform the required blending and send the blended results on to the filters.
2. The correlations in the data from the two filters will be more problematic: without knowledge of the correlations, it will be difficult to properly weight the two data sets, especially the accuracy estimates.
3. The "softish" failure of one of the receivers will be masked by the blending process, resulting in slower failure detection times or possibly even missed failure detections.

If modification to the GPS processing is to be used to reduce CPU usage, selection of the best for processing, with the second used as a standby seems to be the better option, avoiding the problems associated with a blended approach.

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.) Dale Arden Consulting Suite 200, 300 March Road Kanata ON K2K 2E2		2. SECURITY CLASSIFICATION (overall security classification of the document, including special warning terms if applicable) UNCLASSIFIED	
3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C or U) in parentheses after the title.) DIINS Failure Detection, Isolation, and Reconfiguration Procedures (U)			
4. AUTHORS (Last name, first name, middle initial) Arden, Dale A. G.			
5. DATE OF PUBLICATION (month and year of publication of document) August 1997	6a. NO. OF PAGES (total containing information. Include Annexes, Appendices, etc.) 79	6b. NO. OF REFS (total cited in document) 5	
7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.) Final Technical Report			
8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include the address.) Defence Research Establishment Ottawa Ottawa ON K1A 0Z4			
9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant) 1bd11	9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written) W7714-5-9909		
10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique to this document.) DAC-03-97 DRBD CONTRACTOR REPORT 1999-114	10b. OTHER DOCUMENT NOS. (Any other numbers which may be assigned this document either by the originator or by the sponsor)		
11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification) ☒ (X) Unlimited distribution ☐ () Distribution limited to defence departments and defence contractors; further distribution only as approved ☐ () Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved ☐ () Distribution limited to government departments and agencies; further distribution only as approved ☐ () Distribution limited to defence departments; further distribution only as approved ☐ () Other (please specify):			
12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.) UNLIMITED			

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

DCD03 2/06/87

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

The Dual Inertial Integrated Navigation System (DIINS) is a Kalman Filter based, multi-sensor optimally integrated navigation system developed at the Defence Research Establishment Ottawa. DIINS runs a number of filters simultaneously in an architecture called an Hierarchic Alliance of Filters. The filters are arranged in a "root" structure with each of two main branches based on data from one INS. At the top of the root, all available aiding sensors are used to form the measurements. At the base, each filter uses only one aiding sensor. Intermediate filter levels contain ever decreasing numbers of aiding sensors. This architecture ensures that there is always at least one uncorrupted filter, regardless of the sensor failure status. The highest level filter which is safe from corruption by a sensor failure is eligible for use as DIINS's output source.

This report describes two types of sensor failures - hard (an abrupt change in sensor performance identified by measurement tests) and soft (a slowly accumulating change in sensor performance identified by specialized tests). Methods of detecting both failure types are described. Then, procedures are derived which combine the failure detection tests from more than one filter to isolate sensor failures. Finally, failure detection and isolation results are used to select the best source for DIINS output

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Navigation, Kalman Filtering, Inertial Navigation, GPS, Loran-C, error analysis, Failure Detection, Failure isolation, FDIR, Sea trials

#512194